OFFICE OF CIVILIAN RADIOACTIVE WASTE MANAGEMENT
ANALYSIS/MODEL COVER SHEET
Complete Only Applicable Items

1. QA: QA
Page: 1 of: 85

2. ☒ Analysis	Check all that apply	3. ☒ Model	Check all that apply
Type of Analysis	□ Engineering ☒ Performance Assessment ☒ Scientific	Type of Model	☒ Conceptual Model ☒ Mathematical Model ☒ Process Model ☒ Abstraction Model □ System Model
Intended Use of Analysis	□ Input to Calculation ☒ Input to another Analysis or Model ☒ Input to Technical Document ☒ Input to other Technical Products	Intended Use of Model	□ Input to Calculation ☒ Input to another Model or Analysis ☒ Input to Technical Document ☒ Input to other Technical Products
Describe use:	Analyses to be used to identify the range of validity of the particle-tracking model in UZ transport models	Describe use:	Model to be used in the simulation of radionuclide migration in the unsaturated zone

4. Title:

Particle Tracking Model and Abstraction of Transport Processes

5. Document Identifier (including Rev. No. and Change No., if applicable):

ANL-NBS-HS-000026, Rev 00

6. Total Attachments:

4

7. Attachment Numbers - No. of Pages in Each:

I: 6; II: 17; III: 12, IV: 13

	Printed Name	Signature	Date
8. Originator	Bruce Robinson	SIGNATURE ON FILE	4/5/00
9. Checker	Peter Persoff	SIGNATURE ON FILE	4/5/00
10. Lead/Supervisor	Bruce Robinson	SIGNATURE ON FILE	4/7/00
11. Responsible Manager	Clifford K. Ho	SIGNATURE ON FILE	4/7/00

12. Remarks:

Per Section 5.5.6 of AP-3.10Q, the responsible manager has determined that the subject AMR is not subject to AP-2.14Q review because the model does not affect a discipline or area other than the originating organization (Performance Assessment). The downstream user of the model resulting from this AMR is Performance Assessment (PA), which is also the originating organization of this work. PA leads, such as Vinod Vallikat and Patrick Mattie, have worked closely with the originator during the development of the model described in this AMR. Nevertheless, this AMR was sent to various Project participants to give representatives of QA, Regulatory and Licensing, and Applied Research and Testing the opportunity to read the AMR and provide informal comments.

OFFICE OF CIVILIAN RADIOACTIVE WASTE MANAGEMENT
ANALYSIS/MODEL REVISION RECORD

Complete Only Applicable Items

1. QA: QA

Page: 2 of: 85

2. Title:

Particle Tracking Model and Abstraction of Transport Processes

3. Document Identifier (including Rev. No. and Change No., if applicable):

ANL-NBS-HS-000026, Rev 00

4. Revision/Change No.	5. Description of Revision/Change
00	Initial issue

CONTENTS

Page

ATTACHMENTS

FIGURES

Page

TABLES

ACRONYMS

AMR	Analysis and Model Report
CD equation	Convective-dispersion equation
CHv	Calico Hills vitric
CHz	Calico Hills zeolitic
CRWMS M&O	Civilian Radioactive Waste Management System Management and Operating Contractor
DOE	Department of Energy
DTN	Data Tracking Number
EBS	Engineered Barrier System
FEHM	A finite-element heat and mass transfer numerical analysis computer code
GoldSim	The Graphical Simulation Environment software
M&O	Management and Operations Contractor
OCRWM	Office of Civilian Radioactive Waste Management
PA	Performance Assessment
PMR	Process Model Report
QA	Quality assurance
QAP	Quality Assurance Procedure
QARD	Quality Assurance Requirements Document
RTD	Residence Time Distribution
RTTF	Residence time transfer function
SAN	Software Activity Number
SR	Site Recommendation
STN	Software Tracking Number
SZ	Saturated zone
T2FEHM2	Flow Field conversion computer program
TOUGH2	Flow and Transport computer code
TSPA	Total Systems Performance Assessment
TSPA-SR	Total Systems Performance Assessment – Site Recommendation
TSw	Topopah Spring Welded tuff
UZ	Unsaturated zone

VA	Viability Assessment

1. PURPOSE

The purpose of the transport methodology and component analysis is to provide the numerical methods for simulating radionuclide transport and model setup for transport in the unsaturated zone (UZ) site-scale model. The particle-tracking method of simulating radionuclide transport is incorporated into the FEHM computer code and the resulting changes in the FEHM code are to be submitted to the software configuration management system.

This Analysis and Model Report (AMR) outlines the assumptions, design, and testing of a model for calculating radionuclide transport in the unsaturated zone at Yucca Mountain. In addition, methods for determining colloid-facilitated transport parameters are outlined for use in the Total System Performance Assessment (TSPA) analyses. Concurrently, process-level flow model calculations are being carried out in a PMR for the unsaturated zone (CRWMS M&O 2000a). The computer code TOUGH2 (Pruess 1991) is being used to generate three-dimensional, dual-permeability flow fields, that are supplied to the Performance Assessment group for subsequent transport simulations. These flow fields are converted to input files compatible with the FEHM code, which for this application simulates radionuclide transport using the particle-tracking algorithm outlined in this AMR. Therefore, this AMR establishes the numerical method and demonstrates the use of the model, but the specific breakthrough curves presented do not necessarily represent the behavior of the Yucca Mountain unsaturated zone.

The particle-tracking technique presented in this AMR, called the Residence Time Transfer Function (RTTF) particle-tracking technique, uses a cell-based approach that sends particles from node to node on a finite difference or finite element grid, after keeping each particle at the cell for a prescribed period of time. To incorporate transport mechanisms such as dispersion and matrix diffusion, the residence time of a particle at a cell is computed using a transfer function that ensures that the correct distribution of times at the cell is reproduced. This procedure is computationally very efficient, enabling large-scale transport simulations of several million particles to be completed rapidly on modern workstations. This requirement was needed for complex, three-dimensional simulation involving the simulation of multiple radionuclides. Furthermore, since the cell-based approach uses directly mass flow rate information generated from a numerical fluid flow solution, complex, unstructured computational grids and the dual-permeability flow model formulation pose no additional complications. For the present application, the technique is used for unsaturated, dual-permeability transport simulations, for which the method is well suited. The cell-based approach allows accurate simulation of dual-permeability systems in which there is a vast disparity in the travel times depending on whether the transport is in the fractures or the matrix. Furthermore, matrix diffusion and colloid-facilitated radionuclide transport can be simulated. Additionally, complex source terms and decay chain/ingrowth capabilities have been included in the model.

Like all numerical methods, the particle-tracking technique has limitations that must be considered when deciding whether its use is appropriate for a given application. The key physical and chemical assumptions are advection-dominated transport and linear, equilibrium sorption. The possibility of grid orientation effects should also be considered. These effects can manifest themselves as an artificial lateral spreading of solute mass. Also, the accuracy of the method for dual-permeability flow systems was investigated in detail and found to perform best when the flow regime undergoes abrupt transitions at unit interfaces, and in cases for relatively low diffusion. Given these results, this AMR demonstrates that the particle-tracking model can be used in three-dimensional radionuclide transport simulations of the Yucca Mountain unsaturated zone as long as the limits on the model are recognized and parameters are chosen accordingly. However, breakthrough curves presented in this report are not necessarily representative of the Yucca Mountain unsaturated zone. Also, given the accuracy of the model without diffusion, supporting particle-tracking model runs in the absence of diffusion should be performed to access the importance of matrix diffusion to the overall conclusions of the performance assessment.

This analysis is governed by the following Office of Civilian Radioactive Waste Management (OCRWM) Work Direction and Planning Documents: "Improve Documentation and Verification of Radionuclide Decay Model (Rev. 01)" (CRWMS M&O 1999a); "Enhance Particle Tracking Features for TSPA (Rev. 01)" (CRWMS M&O 1999b); "Improve Matrix Diffusion Model (Rev. 01)" (CRWMS M&O 1999c).

2. QUALITY ASSURANCE

The QA program applies to the development of this AMR. The Performance Assessment Operations (PAO) responsible manager has evaluated this activity in accordance with QAP-2-0, *Conduct of Activities*. The QAP-2-0 activity evaluation *Conduct of Performance Assessment* (CRWMS M&O 1999d), has determined that the preparation and review of this technical document of is subject to the *Quality Assurance Requirements and Description (QARD)* DOE/RW-0333P (DOE 2000) requirements. Preparation of this analysis/model did not require classification of items in accordance with QAP-2-3, *Classification of Permanent Items*.

3. COMPUTER SOFTWARE AND MODEL USAGE

Table 1 lists the software and routines used in this study. The computer software code used as a starting point to perform/model the saturated-zone particle tracking in this AMR is FEHM, V2.0, CSCI/ STN# 10031-2.00-00 for a SUN Ultra Sparc, and is under Configuration Management (CM) control. A revised version of the code that includes the particle-tracking algorithm described in this AMR is currently being qualified under AP-SI.1Q, *Software Management*, as FEHM, Version 2.10 (Software Activity Number (SAN) LANL-1999-046, Software Tracking Number (STN): 10086-2.10-00). Version 2.10 is the version used for all calculations performed in this AMR; therefore, the FEHM software is currently unqualified and all results are to be considered TBV. The software is appropriate for the application and was used only within the range for which it was developed. The input and output files for these analyses are being submitted to the Project database for archival as DTN: LA0002BR12213S.002.

Table 1. Computer Software and Routines

Software Name	Version	Software Tracking Number (STN)	Computer Platform
FEHM	2.10	10086-2.10-00	SUN with UNIX OS, FORTRAN
Routines:		**Documentation**	
BKPM	1.0	Attachment II	SUN with UNIX OS, FORTRAN
CHAIN	1.0	Attachment III	PC with Windows NT, FORTRAN
TEST	1.0	Attachment I	SUN with UNIX OS, FORTRAN
test_ebs_random	1.0	Attachment IV	SUN with UNIX OS, FORTRAN

In addition, the following commercially available software was used in this analysis and documentation:

FORTNER PLOT SUN Workstations Version 1.3

This software was used for plotting graphs. Therefore, the software was used for presentation purposes only, and there were no additional routines or macros developed using this commercial software.

4. INPUTS

4.1 DATA AND PARAMETERS

For the code development and testing portions of this AMR, generic values of hydrologic and transport parameters are used that are similar to those to be used in the unsaturated-zone flow and transport modeling activities. However, this code testing does not require referenceable parameter values to verify that the model is correctly implemented. For other site-specific models developed in this AMR, the following data sources were used:

Table 2. Input Data Used

DTN	Data Description
LB990501233129.001	Mean fracture aperture and spacing, variance in aperture
GS980908312242.039 GS950608312231.008 GS960808312231.003	Moisture retention curves – used to calculate pore size distributions
LA0003MCG12213.002	Cumulative Probabilities for Colloid Transport Between One Matrix and Another Calculated from Interpolation of Pore Volume Data from Yucca Mountain Hydrologic (Stratigraphic) Samples
LA0002PR831231.003	Probabilities for constants and retardation factors from C-wells microsphere data.

4.2 CRITERIA

This AMR complies with the DOE interim guidance (Dyer 1999). Subparts of the interim guidance that apply to this analysis or modeling activity are those pertaining to the characterization of the Yucca Mountain site (Subpart B, Section 15), the compilation of information regarding hydrology of the site in support of the License Application (Subpart B, Section 21(c)(1)(ii)), and the definition of hydrologic parameters and conceptual models used in performance assessment (Subpart E, Section 114(a)).

5. ASSUMPTIONS

5.1 GENERAL ASSUMPTIONS OF THE PARTICLE-TRACKING TECHNIQUE

In this section, the assumptions and general approach taken to develop the UZ radionuclide transport model are outlined as the first step toward developing the computational and mathematical models needed in Performance Assessment (PA) calculations.

Prediction of solute transport is a critical element of many groundwater flow studies, including contaminant transport and the movement of natural isotopes or dissolved ions in solution. Modeling efforts typically are motivated by the need to predict the movement of a pollutant or dissolved chemical in the subsurface to answer practical questions concerning the rate and direction of contaminant movement and the predicted concentration in solution. In a typical solute transport simulation, a dissolved chemical is introduced into a steady-state or time-varying flow field, and the fate of the chemical is tracked while undergoing physical and chemical processes such as advection, dispersion, chemical and biological reaction, or diffusion into dead-end pore space. Often, a concentration front is established that must be tracked accurately. In addition, many field investigations employ natural or introduced tracers to study the flow and transport system. These studies also require models to simulate the movement of dissolved species.

Traditional solutions to the advection-dispersion (CD) equation, such as those used in most finite-element or finite-difference flow and transport codes, are versatile and allow the simultaneous solution of multiple interacting species. One drawback of a finite-difference or finite-element solution to the CD equation is that significant numerical dispersion may arise in the portion of a computational domain occupied by a front of rapidly varying concentration. Reducing the numerical dispersion requires either increased grid resolution or higher-order approximation methods, both of which may lead to prohibitive computational costs. Numerical dispersion is identical in character to actual dispersion, so it is difficult to separate numerical from actual dispersion in complex transport simulations.

Approaches to cope with this problem include front-tracking algorithms with multiple grids (e.g., Yeh 1990, Wolfsberg and Freyberg 1994), the method of characteristics (e.g., Chiang et al. 1989), hybrid Eulerian-Lagrangian solution techniques (Neuman 1984), and particle-tracking techniques (e.g., Tompson and Gelhar 1990). Front-tracking algorithms solve the CD equation in integrated form on a numerical grid while tailoring the mesh to increase the resolution of the calculation at fronts. In contrast, an Eulerian-Lagrangian technique casts the CD equation using the total derivative, so that the advection portion of transport can be solved accurately using particle-tracking techniques or the method of characteristics, while the dispersion component of transport is solved on a finite-difference or finite-element grid using standard techniques.

Particle-tracking transport models take a fundamentally different approach. The trajectory of individual molecules or packets of fluid containing molecules are tracked through the model domain. When the fluid path lines are the model result of interest (Pollack, 1988; Lu, 1994), a relatively small number of particles can be used to trace the streamlines. Particle tracking is also used to simulate solute transport, such as the migration of a contaminant plume (Akindunni et al. 1995) or the prediction of breakthrough curves in interwell tracer experiments (Johnson et al. 1994). For these applications, a relatively large number of such particles must be used to obtain accurate solutions to the transport problem. Particle tracking has also been used to solve the advective portion of complex reactive transport models that simulate chemical reactions among multiple species (Fabriol et al. 1993).

In a typical particle-tracking algorithm, a particle is sent to a new position assuming that the magnitude and direction of the velocity vector are constant during a time step . If small enough time steps are taken, particle pathways can be tracked accurately. Dispersion is treated as a random process that diverts the particle a random distance from its dispersion-free, deterministic path. In these so-called “random walk” models (e.g., Kinzelbach 1988), dispersion is usually calculated stochastically subject to a Gaussian model to reproduce the specified dispersion coefficient. The technique has also been extended by employing non-Gaussian random walk functions to represent scale-dependent dispersion (Scheibe and Cole 1994). Linear equilibrium sorption can be handled through the use of a retardation factor to correct the magnitude of the particle velocity.

A crucial component of most random-walk particle-tracking algorithms developed to date is the need to accurately estimate the velocity at every position in the model domain. In the context of a finite-difference or finite-element numerical code, this means that velocities at positions other than the node points of the fluid flow grid must be computed using an interpolation scheme. Many studies have proposed and studied the accuracy of different interpolation schemes, including methods developed for regular, two- or three-dimensional finite difference grids (Schafer-Perini and Wilson 1991, Zheng 1993), for two- and three-dimensional finite-element grids (Cordes and Kinzelbach 1992), and for codes that employ the boundary element method for computing fluid flow (Latinopoulos and Katsifarakis 1991). Special techniques have been developed to handle complexities such as point fluid sources and sinks and abrupt changes in the conductivity of the medium (Zheng 1994).

Unfortunately, many of the velocity interpolation schemes used in conventional particle-tracking techniques are computationally intensive, thus limiting the number of particles that can practically be used. Another drawback to traditional particle-tracking approaches is that spatial and temporal discretization often results in numerical inaccuracy in the fluid flow solution upon which velocity determinations are based. Thus, precise and time-consuming velocity interpolation schemes may not be justified in finite-difference or finite-element models. Finally, and most important for the simulation of transport in the UZ at Yucca Mountain, dual-permeability models employ overlapping continua to represent fracture and matrix flow

(Zyvoloski et al. 1992, Zimmerman et al. 1993). To develop a streamline-based particle-tracking method for dual-permeability models, velocity interpolations on each continuum would have to be coupled to a transfer term that allows particles to move from one medium to the other. This additional complexity, along with the inherent approximations associated with the dual-permeability method itself, may make precise velocity interpolation calculations of limited validity.

In this AMR, a new particle-tracking technique is developed for transient, multi-dimensional finite-difference or finite-element codes. The algorithm is designed for computing solute concentration fields quickly and easily with structured or unstructured numerical grids of arbitrary complexity. Both continuum and dual-permeability formulations can be simulated. This flexibility is accomplished by extending the cell-based strategy of Desbarats (1990) for mapping out the path of the particle. In this method, the calculation of an "exact" pathline is replaced with a cell-to-cell migration of the particle. The mass flux from cell to cell is used directly, and no velocity interpolations are required. Since numerical solutions for fluid flow are typically mass-conservative (though not necessarily accurate) the particle-tracking method automatically conserves mass.

In subsequent sections, the mathematical basis for this algorithm is outlined, and theory is developed to incorporate the effects of sorption, dispersion, and matrix diffusion into this new particle-tracking framework. Then, the technique is verified by comparing the results to a variety of analytical solutions and by benchmarking the code against a finite-element solution of the advection-dispersion for a one-dimensional, dual-permeability flow field. Finally, specific capabilities are developed for the radionuclide transport simulations, including source term methods, decay chains/ingrowth options, and colloid-transport methods.

5.2 ASSUMPTIONS OF THE ACTIVE FRACTURE MODEL FOR DETERMINING FRACTURE SPACING AND APERTURE

All assumptions stated in this section are used in Section 6.2.1.

1. Fracture frequency, aperture, and permeability are log-normally distributed.

Basis: These properties are a quantities bounded at the low end by zero. Therefore, a log-normal distribution is a natural choice that can meet the measured means and standard deviations, and are constrained to be larger than zero. This assumption does not require verification.

2. The cubic law is a valid approximation for gas permeability in fractured rock at Yucca Mountain.

Basis: The cubic law for fracture permeability is a basic relationship used to relate fracture properties and fracture permeability (Bear et al. 1993, p. 15). This is an adequate approximation, based on scientific judgement, for the purposes of establishing the fracture aperture variance. No verification is required.

3. Active fracture model appropriately accounts for reduced fracture/matrix interaction.

Basis: The reduction in fracture/matrix contact area is a result of the active fracture unsaturated flow model. This reduction is justified on the basis of the desirability of maintaining consistency with the assumptions underlying the development of the flow fields developed for the UZ flow modeling effort. These assumptions are developed in Liu et al. (1998). No further confirmation of this assumption is required.

5.3 ASSUMPTIONS OF THE COLLOID TRANSPORT MODEL

All assumptions stated in this section are used in Section 6.1.4.

1. Radionuclide mass sorbs reversibly to non-diffusing colloids with a constant equilibrium sorption parameter $K_c = C_{coll} / C_{fluid}$, where C_{coll} is the radionuclide concentration residing on the colloids (moles radionuclide on colloid per kg fluid), and C_{fluid} is the corresponding concentration in the fluid phase (moles aqueous radionuclide per kg fluid).

Basis: Most measurements of sorption onto bulk rock and colloids are interpreted using an equilibrium sorption model such as this one. For compatibility with the data collected on sorption, this assumption is adopted in the numerical model as well. If attachment to colloidal particles is thought to be truly irreversible, this model can be used with an extremely large value of K_c for that portion of the radionuclide inventory. This assumption does not require verification.

2. Colloids undergo reversible filtration in the porous medium, with a colloid retardation factor of R_{coll}.

Basis: To estimate retardation of colloids in the fracture continuum, field experiments at the C-wells complex near Yucca Mountain were examined, in which transport of microspheres was used as an analog for colloids. The microsphere breakthrough curves were fit to forward and reverse filtration rates (DTN: LA0002PR831231.003). These rate constants were then used to calculate a retardation factor for colloid transport through saturated fractured rock (Table 7 of this document, from CRWMS M&O 2000b). For compatibility with this analysis of field data, this assumption is adopted in the numerical model as well. This assumption does not require verification.

3. Colloids undergo dispersion with identical dispersivity as an aqueous solute.

Basis: When dispersivity is used to model solute spreading in porous media, it is introduced to capture variability in the flow velocity that exists at smaller scales than are modeled in the numerical grid. To a first approximation, this variability will act similarly on aqueous and colloidal components. Therefore, the same dispersivity should be used for both. This assumption does not require verification.

6. ANALYSIS/MODEL

6.1 THE RTTF PARTICLE-TRACKING TECHNIQUE

This section outlines the development of the general transport methods used for the RTTF particle-tracking model, and Section 6.2 discusses issues specific to the use of this model to simulate radionuclide transport for the Yucca Mountain UZ.

6.1.1 Basic Methods

The particle-tracking method developed in the present study views the fluid flow computational domain as an interconnected network of fluid storage volumes. Particles travel only from cell to cell, requiring no greater resolution of the particle pathways. In this sense, the method is similar to the node-to-node routing method of Desbarats (1990, p. 156). This simple starting point has been extended to include many different transport submodels and complex flow domains. The description that follows is applicable for steady-state, single-porosity flow fields; the corrections to the method for treating transient flow systems and dual-porosity model formulations are discussed in subsequent sections. The two steps in the particle-tracking approach are: 1) determine the time a particle spends in a given cell; and 2) determine which cell the particle travels to next. These two steps are detailed below.

The residence time for a particle in a cell is governed by a transfer function describing the probability of the particle spending a given length of time in the cell. Thus, this particle-tracking approach is called the “residence time/transfer function” (RTTF) method. The schematic plots shown in Figures 1 and 2 illustrate the basis of the RTTF approach. For a cumulative probability distribution function of particle residence times, the residence time of a particle in a cell is computed by generating a random number between 0 and 1 to determine the corresponding residence time from the distribution function. In this example, the advection-dispersion equation was used to generate the RTTF curve, but other transport mechanisms can be incorporated as well, as demonstrated below.

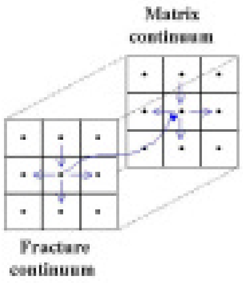


Figure 1. Schematic of the Cell-Based Particle-tracking Technique

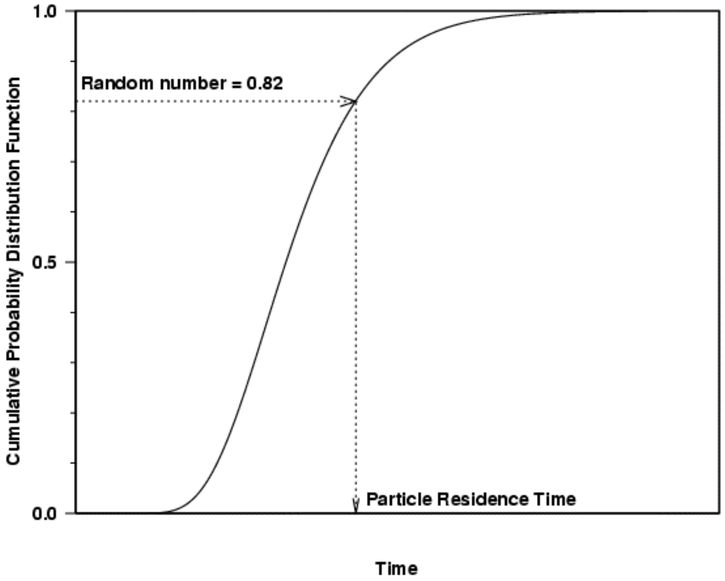


Figure 2. Schematic of the RTTF Technique for Determining Particle Residence Time in a Cell

If a large number of particles pass through the cell, the cumulative residence time distribution (RTD) of particles in the cell will be reproduced. Particle-tracking models of single-fracture transport (Yamashita and Kimura, 1990) have employed this approach to simulate fracture transport of diffusion into the rock matrix. From the solution of the flow field in a numerical model, the mass of fluid in the computational cell and the mass flow rate to or from each adjacent cell is computed. In the simplest case, the residence time of a particle in a cell, τ_{part}, is given by

$$\tau_{part} = \tau_f = \frac{M_f}{\sum \dot{m}_{out}} \qquad \text{(Eq. 1)}$$

where M_f is the fluid mass in the cell and the summation term in the denominator refers to the outlet fluid mass flow rates from the cell to adjacent cells. In the absence of dispersion or other transport mechanisms, the transfer function describing the distribution of particle residence times is a Heaviside function (unit step function) that is unity at the fluid residence time τ_f, because for this simple case, all particles entering the cell will possess this residence time. Equilibrium, linear sorption is included by correcting the particle residence time by a retardation factor R_f. Thus, $\tau_{part} = R_f \tau_f$, and R_f is given by

$$R_f = 1 + \frac{\rho_b K_d}{\phi S_f} \qquad \text{(Eq. 2)}$$

where

K_d is the equilibrium sorption coefficient (mL fluid/g rock)

ρ_b is the bulk rock density (g rock/mL total)

ϕ is the porosity(mL pore space/mL total)

S_f is the saturation of the phase in which the particle is traveling (mL fluid/mL pore space).

Once again, in the absence of other transport processes, the transfer function is a Heaviside function.

Before discussing more complex transfer functions for the RTTF method, the method for determining which cell a particle travels to after completing its stay at a given cell is discussed. The approach that is consistent with the RTTF method is that the probability of traveling to a neighboring cell is proportional to the mass flow rate to that cell. Only outflows are included in this calculation; the probability of traveling to an adjacent node is 0 if fluid flows from that node to the current node. A uniform random number from zero to one is used to make the decision of which node to travel to. Thus, the particle-tracking algorithm is: 1) compute the residence time of a particle at a cell using the RTTF method; and 2) at the end of its stay, send the particle to an adjacent cell randomly, with the probability of traveling to a given cell proportional to the mass flow rate to that cell.

6.1.2 Dispersion

Transport processes such as dispersion can be incorporated into the RTTF particle-tracking algorithm through the use of transfer functions. For dispersion, within a computational cell, the equation for one-dimensional, axial dispersion is applied. The transport equation and boundary conditions for the one-dimensional, advective-dispersion equation are:

$$R_f \frac{\partial C}{dt} = D_{eff} \frac{\partial^2 C}{dz^2} - v \frac{\partial C}{dz} \qquad \text{(Eq. 3)},$$

$$C = 0, \qquad t = 0 \qquad \text{(Eq. 4)},$$

$$C = C_0, \qquad z = 0 \qquad \text{(Eq. 5)},$$

$$C = 0, \qquad z \to \infty \qquad \text{(Eq. 6)},$$

where

C is the concentration (moles/kg fluid)

C_0 is the injection concentration (moles/kg fluid)

v is the flow velocity (m/s)

D_{eff} is the effective dispersion coefficient (m^2/s), given by $D_{eff} = \alpha v$, where α is the dispersivity of the medium (m).

Here the molecular diffusion coefficient is ignored, since in general it is much smaller than the flow dispersion component of D_{eff}. A non-dimensional version of Equation 3 can be obtained by the following transformations: $\widehat{C} = C / C_0$, $\widehat{z} = z / L$, and $\theta = vt / R_f L$, where L is the flow path length. The solution to Equations 3-6 is obtained after manipulation of Freeze and Cherry (1979, p. 391, Equation 9.5), yielding:

$$\widehat{C} = \frac{1}{2}\left[erfc\left(\frac{\sqrt{Pe}(1-\theta)}{2\sqrt{\theta}} \right) + e^{Pe} erfc\left(\frac{\sqrt{Pe}(1+\theta)}{2\sqrt{\theta}} \right) \right] \quad \text{(Eq. 7)}$$

where Pe is the Peclet number (dimensionless), $Pe = vL / D_{eff} = L / \alpha$.

The use of this solution in the RTTF particle-tracking method requires that the transport problem be advection-dominated, such that during the time spent in a computational cell, solute would not tend to spread a significant distance away from that cell. Then, the approximate use of a distribution of times within a single cell will be adequate. Quantitatively, the criterion for applicability is based on the grid Peclet number $Pe_g = \Delta x / \alpha$, where Δx is the characteristic length scale of the computational cell. Note that in contrast to conventional numerical solutions of the advective-dispersion equation, coarse spatial discretization is helpful for satisfying this criterion. Of course, the mesh spacing must still be small enough to provide an accurate flow solution. Highly dispersive transport invalidates the assumptions of the RTTF particle-tracking technique. When dispersion coefficients are large, accurate solutions to the advective-dispersion equation are easily obtained by conventional finite-difference or finite-element techniques, so these techniques should be used instead under these circumstances.

For multi-dimensional flow systems, the dispersion model developed for one-dimensional systems can be extended to include dispersion coefficient values aligned with the coordinate axes. For this case, the flow direction is determined by the vector drawn from the nodal position of the previous cell to the current cell, and the dispersivity for this flow direction is computed from the equation for an ellipsoid:

$$\alpha = \frac{L}{\sqrt{\Delta x^2 / \alpha_x^2 + \Delta y^2 / \alpha_y^2 + \Delta z^2 / \alpha_z^2}} \quad \text{(Eq. 8)}$$

The RTTF particle-tracking technique cannot be simply formulated with a longitudinal and transverse dispersion coefficient model, with the tensor aligned with the flow direction, because the flow rates between cells are defined rather than the actual flow velocity at a position. For a dispersion model aligned with the flow direction, a random-walk particle-tracking method such as that of Tompson and Gelhar (1990), also implemented in the saturated zone particle-tracking algorithm of FEHM, or a conventional finite-element or finite-difference solution to the CD equation, such as the reactive transport solution module in FEHM, should be used instead.

The numerical implementation of this technique requires the determination of the dimensionless time θ in Equation 7 for a randomly determined value of the dimensionless concentration $\widehat{C}$.

This determination is accomplished numerically in the particle-tracking code by fitting Equation 7 at selected values of θ between 1 and 1000 using a piecewise continuous series of straight lines spanning the entire range of values. Then, the value of θ at an arbitrary value of Pe is computed by linear interpolation between values determined at the Peclet numbers that bracket the actual value. This technique, involving a simple search for the correct type curves, followed by the calculation of two values of θ and an interpolation, is much more computationally efficient (about a factor of two in cpu time) and robust than an iterative approach to the exact solution using Newton's method. Solutions of adequate accuracy (less than 1% error) for Peclet numbers between 1 and 1000 are easily obtained using this linear-interpolation method.

6.1.3 Matrix Diffusion

Matrix diffusion has been recognized as an important transport mechanism in fractured porous media (e.g., Neretnieks 1980; Robinson 1994). For many hydrologic flow systems, fluid flow is dominated by fractures because of the orders-of-magnitude larger permeabilities in the fractures compared to the surrounding rock matrix. However, even when fluid in the matrix is completely stagnant, solute can migrate into the matrix via molecular diffusion, resulting in a physical retardation of solute compared to pure fracture transport. This effect has recently been demonstrated on the laboratory scale by Reimus (1995) and on the field scale both by Maloszewski and Zuber (1991) and in the saturated zone at Yucca Mountain by Reimus et al. (1999).

To derive a transfer function for matrix diffusion, an idealized representation of the transport system must first be developed. In this particle-tracking algorithm, the model depicted in Figure 3 is used to provide a transfer function for the case of fracture flow and diffusion between equally spaced fractures. In this model, additional terms not defined earlier are as follows:

t is the time (s)
$2b$ is the fracture aperture (m)
$2B$ is the mean fracture spacing (m)
ϕ is the porosity of the matrix
$R_{f,m}$ is the retardation factor in the matrix
$R_{f,f}$ is the retardation factor in the fracture
D is the effective diffusion coefficient (m^2/s)

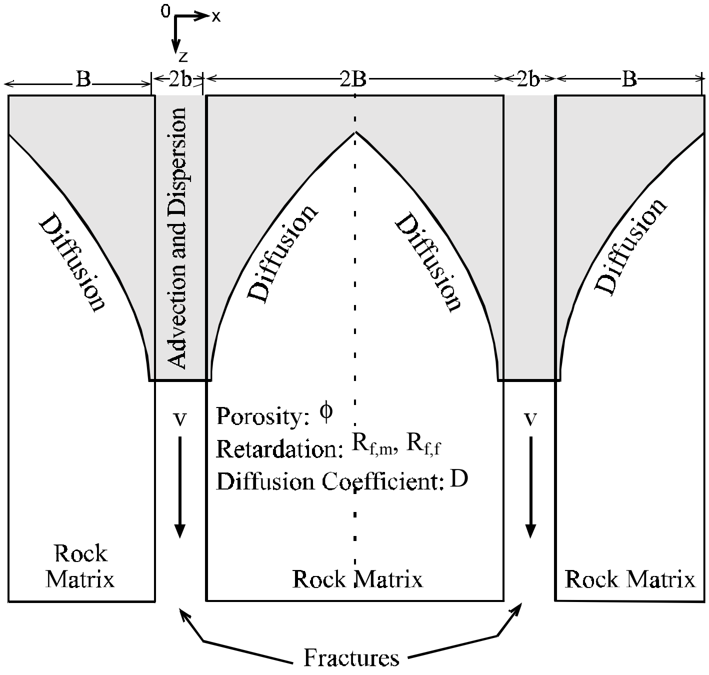


Figure 3. Schematic of the Matrix-Diffusion Submodel

Transport in the fractures is governed by Equation 3 with an additional term describing diffusion from the fracture to the matrix:

$$R_{f,f}\frac{\partial C}{dt}=D_{eff}\frac{\partial^2 C}{dz^2}-v\frac{\partial C}{dz}-\frac{q}{b} \quad \text{(Eq. 9)}$$

where the flux at the fracture/matrix interface is given by

$$q=-\phi D\frac{\partial C}{dx}\bigg|_{x=b} \quad \text{(Eq. 10)}$$

and transport within the matrix is described by the one-dimensional diffusion equation

$$R_{f,m}\frac{\partial C}{\partial t}=D\frac{\partial^2 C}{\partial x^2} \quad \text{(Eq. 11),}$$

The molecular diffusion coefficient is the product of the free diffusion coefficient of the solute in water and a tortuosity factor to account for the details of diffusion through a tortuous, fluid-filled pore network. In this model, D is treated as the fundamental transport parameter, recognizing that it is a property of both the solute and the medium.

Although a particular boundary condition in between the fractures is depicted in Figure 3, there are a variety of analytical solutions to this transport problem depending on the nature of the boundary condition in the x-direction. An analytical solution is given by Tang et al. (1981) for the semi-infinite boundary condition $\partial C/\partial x=0$ as $x\rightarrow B$. For the case of plug flow (no dispersion) in the fractures, the solution of Starr et al. (1985, Equation 8b) can be used. Replacing Starr's term x / v_s with τ_f, as is called for in the RTTF technique, and ignoring radioactive decay, their solution reduces to

$$\frac{C}{C_0} = erfc\left[\frac{\phi \tau_f \sqrt{R_{f,m} D}}{2b\sqrt{t - R_f \tau_f}}\right] \qquad \text{(Eq. 12)}$$

for $t > R_f \tau_f$, and $C / C_0 = 0$ for $t \leq R_f \tau_f$. The semi-infinite boundary condition between fractures limits the validity of either of these solutions to situations in which the characteristic diffusion distance for the transport problem is small compared to the fracture spacing B. As long as the solute has insufficient time to diffuse to the centerline between fractures, the solution provided by Tang et al. (1981) can be used as the transfer function for the particle-tracking technique. Alternatively, the boundary condition depicted in Figure 3 can be used to provide a transfer function for the case of fracture flow and diffusion between equally spaced fractures. Under these conditions, the analytical solution provided by Sudicky and Frind (1982) can be used to obtain the transfer function. The derivation of a form of this solution suitable for incorporation into the particle-tracking methodology has been presented in CRWMS M&O (2000c) for the saturated zone (SZ) particle-tracking transport model. The same subroutines in FEHM are used for both the UZ and SZ transport models.

Although in principle a solution such as Tang et al. (1981) that includes dispersion and matrix diffusion could be used directly for the transfer function, its complex form makes it very inconvenient for rapidly computing particle residence times. Instead, a two-step process is used wherein the residence time in the fracture is first computed using the transfer function for one-dimensional dispersion (Equation 7) without sorption. This fracture residence time is then used in the plug-flow equation with matrix diffusion and sorption to compute the particle residence time. To use Equation 12 as a transfer function, a numerical algorithm was developed to determine the inverse of the error function, that is, the value of x_d for a given value of y_d, such that $y_d = erf(x_d)$ (note that $erfc(x_d) = 1 - erf(x_d)$). The numerical implementation of this method entails dividing the error function into piecewise continuous segments from which the value of x_d is determined by interpolation. The use of the two-step approach is justified because of the principle of superposition, which allows the dispersive process in the fracture to be decoupled from diffusive transport in the matrix. Proof that this numerical technique is acceptable is presented in the Code Verification section (Section 6.3) of this document.

For the finite fracture spacing model, there are two options for defining the fracture spacing. The first is a simple node-by-node assignment of the fracture spacing. The second, the so-called "active fracture model," is more consistent with the UZ flow model and, thus, is outlined in detail in Section 6.2.1.

6.1.4 Colloid Transport

For colloid-facilitated transport, the transport equations for matrix diffusion, with either the semi-infinite or finite fracture spacings, can be simply revised. Given the assumptions listed in Section 5.3, the expression for transport for contaminant on colloids is:

$$R_{coll}\frac{\partial C_{coll}}{dt} = D_{eff}\frac{\partial^2 C_{coll}}{dz^2} - v\frac{\partial C_{coll}}{dz} \qquad \text{(Eq. 13)}$$

where D_{eff} is the same as for an aqueous solute (assumption 3, Section 5.3). Combining Equations 13 and 9, and making use of the relation $K_c = C_{coll} / C_{fluid}$:

$$\left(\frac{R_f + K_c R_{coll}}{1+K_c}\right)\frac{\partial C}{dt} = D_{eff}\frac{\partial^2 C}{dz^2} - v\frac{\partial C}{dz} - \frac{q}{b(1+K_c)} \qquad \text{(Eq. 14)}$$

This derivation implies that the transport equation for matrix diffusion can be revised to include colloid facilitated transport by replacing the half-aperture *b* by

$$b_{coll} = (1+K_c)b \qquad \text{(Eq. 15)}$$

And the retardation factor in the fracture by

$$R_{f,coll} = \frac{R_f + K_c R_{coll}}{1+K_c} \qquad \text{(Eq. 16)}$$

Alternatively, for transport in a porous continuum, the solution to Equation 3 is used, with the retardation factor given by Equation 16, and R_f replaced by $R_{f,m}$. These relationships are built into the FEHM particle-tracking code, so that the additional terms K_c and R_{coll} are provided as inputs.

In addition to the transport of radionuclides bound to colloids, there are several mechanisms related to the migration of the colloids themselves that can be simulated in the model. Above, the reversible retardation factor for colloid migration R_{coll} was introduced. Due to the colloid size and surface properties relative to the pore size and surface charge, colloids can also undergo size exclusion and/or filtration in porous media. In the particle-tracking module, models have been implemented for these processes. For advective flow from fracture to matrix in the dual-permeability model, a size-exclusion model is implemented whereby colloids can remain in the fracture in proportions greater than the relative flow rate entering the matrix. A size exclusion parameter $f_{coll} \leq 1$ is defined such that the probability of particles entering the matrix due to advective transport is multiplied by this factor. Therefore, complete exclusion from the matrix is obtained by setting $f_{coll} = 0$, whereas aqueous solute behavior is retained by setting $f_{coll} = 1$. Filtration, resulting in complete immobilization of the particle, can also be simulated at specified interfaces within either the fracture or matrix continua. To invoke this mechanism, a filtration factor f_{filt} at an interface (the finite element connections between two specified zones in an FEHM simulation) is defined. If a particle is slated to pass from one zone to the other via advective transport, f_{filt} is the probability the particle continues moving, $(1 - f_{filt}$ is the probability that it is irreversibly removed by filtration).

When using the filtration option, a word of caution is warranted. Colloid simulations are typically used to provide a mechanism for radionuclides to travel in the water bound to colloids. Filtration renders the colloids immobile, which in reality, only renders the radionuclide immobile if it is irreversibly bound to the colloid. When the radionuclide is only weakly sorbed to the colloid, the filtration option will artificially remove radionuclide mass from the system, resulting in a non-conservative simulation. Therefore, the filtration option should only be invoked for irreversibly bound radionuclides or when simulating colloid tracer experiments. The reversible model, using R_{coll} to delay the migration, should be used instead for colloid-facilitated transport of radionuclides.

6.1.5 Particle Sources and Sinks

There are two methods for introducing particles into the flow system: (1) the particles are either injected with the source fluid entering the model domain or (2) released at a particular cell or set of cells. The first method is used to track source fluid as it passes through the system. The number of particles entering with the source fluid at each cell is proportional to the source flow rate at that cell, which is equivalent to injecting fluid with a constant solute concentration. For Method 2, an arbitrary number of particles are released at each specified cell, regardless of the source flow rate. In the present application, Method 2 is used to input particles, which are used to represent radionuclide mass into the system at the repository level.

Within Method 2, there are various ways to input a time-varying source of particles. For stand-alone simulations, the particles are inserted at a constant rate for a specified duration. There is also an option, used when the FEHM code is interfaced with GoldSim, to input a time-varying and spatially varying source mass flux into the model. The details of the method for accepting complex sources of multiple radionuclides from the EBS model are discussed in a subsequent section.

When fluid exits the model domain at a sink, the model treats this flow as another outlet flow from the cell. The decision of whether the particle leaves the system or travels to an adjacent cell is then made on a probabilistic basis, just as though the fluid sink were another connected cell. Thus, the complexities discussed by Zheng (1994) for handling a so-called weak sink are avoided in the RTTF particle-tracking model.

6.1.6 Transient Fluid Flow

When the RTTF particle-tracking method is implemented for a time-varying fluid flow system, the approach is somewhat more complex but still tractable. Consider a numerical simulation in which a discrete time step is taken at time t and a new fluid flow field is computed. In this model, the new fluid flow time $t_{new} = t + \Delta t$ is treated as an intermediate time at which the particle-tracking calculation must stop. The time is intermediate because if the flow field is at steady state, there is no reason to stop at any time before the end of the simulation except to record particle information for output or processing purposes. The fate of all particles is tracked from time t to time t_{new} assuming that the flow field is constant over this time interval. When the simulation reaches t_{new}, the position (cell number) of the particle is recorded, along with its fractional time remaining at the cell and the randomly generated y-coordinate of the transfer function used for that particle in the cell. When the new fluid flow solution is established, the remaining residence time for a particle is determined from the following steps:

1. Compute a new fluid residence time τ_f.

2. Using the *y*-coordinate of the transfer function previously computed and the new transfer function, calculate a new particle residence time.

3. Multiply this time by the fractional time remaining in the cell to obtain the remaining time in the cell.

This method approximates the behavior in a transient system, while reducing to the behavior that would be obtained in an unchanging flow field had the calculation not been forced to stop at the intermediate time.

Another transient effect that must be considered is that the sum of the outlet mass flow rates $\sum \dot{m}_{out}$ in Equation 1 does not necessarily equal the sum of the inlet mass flow rates. When there is net fluid flow into a cell, the particle-tracking algorithm uses the sum of the inlet flow rates in Equation 1, whereas Equation 1 itself is used when there is net outflow from a cell.

6.1.7 Dual-Permeability Formulation

In the development of the dual-permeability flow model, the Yucca Mountain project recognized the need to explicitly account for fracture flow in some hydrologic units. A dual-permeability formulation allowed for a greater liklihood of fracture flow by relaxing the assumption inherent in the equivalent continuum model formulation that the fracture and matrix media must be in capillary pressure equilibrium at all locations. In the dual-permeability models, significant fracture flow occurs in units with low matrix permeability such as the Topopah Spring and zeolitic Calico Hills units. This advance allowed the project to reconcile the infiltration rates estimated in surface and soil moisture studies with the fact that in many units, matrix rocks were at less than complete saturation. In an equivalent continuum model formulation, this observation implied infiltration rates that were much lower than values estimated by more direct means. In the dual-permeability model, fractures are able to flow even though there is a high capillary pressure in the matrix blocks.

Because of the success of dual-permeability models in their ability to simulate fracture flow, it would be tempting to use the same model formulation for radionuclide transport. Although this approach would have the advantage of simplicity and consistency with the flow model, there may be significant limitations that would render a simple dual-permeability transport formulation less than adequate for radionuclide transport modeling. The restriction of a single matrix node for each fracture node means that sharp concentration gradients away from the fracture cannot be captured. For flow modeling, this may not be overly restrictive because of the generally more rapid migration of a pressure front into the matrix. Furthermore, the typical scenario of flow modeling is the simulation of fluid velocities, pressures, and saturations under the assumption of steady state flow. By contrast, transport of a contaminant plume is inherently a transient problem, and characteristic diffusional distances can be of the order of centimeters during unsaturated transport through a fractured medium. If the characteristic spacing between flowing fractures is much greater than centimeters, then the use of one grid point in the matrix is questionable for capturing the fracture/matrix diffusion term in the transport mass balance.

Clearly, the biggest advantage of the dual-permeability model formulation for transport is the ability to capture, in a computationally efficient manner, the disparate flow velocities in the two media. However, because of the potential limitations of the dual-permeability formulation for transport, a hybrid model was developed using the cell-based particle-tracking algorithm and the RTTF approach. This method addresses the issue of potentially sharp gradients away from the fractures while relying on the dual-permeability flow formulation to adequately capture the distribution of advective velocities within the model domain. Extension of the advection term in the RTTF particle-tracking technique to handle dual-permeability systems is straightforward, because the fracture-matrix flow term is simply an additional inlet or outlet flow rate from the node. In the algorithm, this flow term is treated like flow to any other node, and the particle can shift from one continuum to the other. In addition, to simulate molecular diffusion as an additional fracture-matrix transport interaction term, the matrix diffusion option can be invoked for particles traveling in the fracture continuum. In this formulation, the simplification employed is that for the purposes of capturing the matrix diffusion component of the transport, particles delayed by matrix diffusion in the fracture continuum remain in the fracture continuum. Advective motion can drive particles from one continuum to the other, as often occurs in dual-permeability models with units of contrasting hydrologic properties. However, the matrix diffusion option assumes a model system in which the fracture and matrix flow systems are weakly coupled, such that matrix diffusion in a fracture can be simulating without explicitly accounting for the superimposed effect of advection.

Of course, the method for modeling solute diffusion between the fractures and matrix is approximate, and needs to be tested against a model that more precisely captures the transport behavior. In Section 6.4, a series of dual-permeability benchmarking calculations are presented to explore the adequacy of the model for the purpose of simulating radionuclide migration in the Yucca Mountain unsaturated zone and to highlight potential strengths and limitations of the model.

6.1.8 Output Options

There are several methods for obtaining output results for a particle-tracking model simulation. Generally, the results are reported in the form of a breakthrough curve, that is, a distribution of arrival times of particles at a fluid sink. These sinks can be lumped together, so that, for example, the breakthrough curve can be related to a mass flux of radionuclide at the water table in the UZ transport model. The sink nodes can also be sub-divided into user-defined zones, and breakthrough curves can be obtained for each.

In addition to breakthrough curves, in situ concentrations can be calculated and reported at a given node or nodes, or throughout the model domain. However, as is the case with all particle-tracking algorithms, concentration distributions can be subject to significant noise when used to compute concentration distributions. This problem can be especially acute for dual-permeability systems, for which transport velocities in the fractures can be so rapid that the probability of finding a statistically significant number of particles in a fracture grid node at a particular reporting period is very low. Alternatively, a provision in the code allows the cumulative number of particles passing through a cell to be reported. When a pulse input of particles is used, this type of concentration is equivalent to the response to a constant mass flux input of solute. Since all particles passing through a cell get counted, the problem just described is alleviated.

6.2 IMPLEMENTATION ISSUES FOR THE UZ TRANSPORT MODEL

This section contains issues specifically related to the use of the particle-tracking model for the Yucca Mountain UZ.

6.2.1 Active Fracture Model

The transport equation for a single fracture under unsaturated conditions can be obtained by a simple extension of Equation 9:

$$R_{f,f}\frac{\partial C}{dt}=D_{eff}\frac{\partial^2 C}{dz^2}-v\frac{\partial C}{dz}-\frac{R_A q_{fm}}{S_f b} \qquad \text{(Eq. 17)}$$

Here

S_f is the fracture saturation

q_{fm} is the fracture/matrix diffusive flux (moles/m^2-s)

R_A is the fracture/matrix contact area reduction factor.

The term $S_f b$ may be thought of as the “water aperture” for the transport problem. In the active fracture model, only a portion of the fracture space participates in the flow and transport. The saturation of the active portion is the “active fracture saturation”. It is this saturation that applies to the transport equation because the inactive fractures can be thought of as unavailable, as if filled with minerals. Therefore, S_f is replaced by the active fracture saturation, S_a. The fracture/matrix area reduction factor is defined by Liu et al. (1998; Equation 13) to be:

$$R_A = S_{ae} \qquad \text{(Eq. 18)}$$

where S_{ae} is the effective water saturation of the active fractures. The effective (or normalized) water saturation is given by Liu et al. (1998, Equation 4):

$$S_{ae} = \frac{S_a - S_r}{1 - S_r} \tag{Eq. 19}$$

where S_r is the residual water saturation. This area reduction factor was derived for unsaturated flow, for which it can be argued that the effective fracture/matrix flow process goes to zero at residual water saturation. For matrix diffusion and transport, however, the effective contact area should only go to zero when the physical saturation of the active fractures goes to zero. Therefore,

$$R_A = S_a \tag{Eq. 20}$$

Substituting $S_f = S_a$ and $R_A = S_a$ into the transport equation:

$$R_{f,f} \frac{\partial C}{dt} = D_{eff} \frac{\partial^2 C}{dz^2} - v \frac{\partial C}{dz} - \frac{q_{fm}}{b} \tag{Eq. 21}$$

which is the same equation as for transport in a saturated fracture.

The adjustment of the fracture spacing to obtain the fracture spacing of active fractures is given by the following relationship (adapted from Liu et al., 1998, Equation 17):

$$B = B_g \widehat{S}^{-\gamma} \tag{Eq. 22}$$

where

$$\widehat{S} = \frac{S_f - S_r}{1 - S_r} \tag{Eq. 23}$$

B_g is the geometric fracture spacing (including flowing and dry fractures)

S_r is the residual fluid saturation

exponent γ is a fitting parameter ($0 < \gamma < 1$).

The model already has a local value of S_f at every node, and thus requires B_g, S_r, and γ as input parameters.

In this section, the UZ parameter distributions to be used for the fracture aperture and spacing are derived for use in the finite fracture spacing matrix diffusion model. The inverse of the fracture aperture is used in the transport model as a measure of the amount of fracture surface area per unit volume of fracture pore space. Given that the fracture area per unit bulk volume, *A*, is available from the flow model, as listed in Table 3, and that the fracture pore volume per unit bulk volume (or fracture porosity, ϕ_f) is also given in Table 3, it is a simple matter to compute the fracture aperture, *b*, as follows:

$$b = \frac{\phi_f}{A} \qquad \text{(Eq. 24)}$$

The aperture derived from this formula will be used as the geometric mean of the distribution for aperture. The relationship for fracture spacing is simply the inverse of fracture frequency.

Radionuclide transport is not expected to be very sensitive to fracture spacing. However, it is expected to be sensitive to fracture aperture. Therefore, a stochastic sampling for fracture aperture in the TSPA calculations is desired. Although the analysis above provides a simple means to evaluate the mean fracture aperture, the data for area and porosity do not lend themselves to the evaluation of a variance in fracture aperture for the purposes of sampling. However, using the data in Table 4, it is possible to establish a variance in fracture aperture by the following method. From the cubic law, the fracture aperture $2b$ may be expressed as:

$$2b = \left(\frac{12\, k_G}{f} \right)^{\frac{1}{3}} \qquad \text{(Eq. 25)}$$

where k_G is the permeability (m^2) of the fractured medium determined from estimates of gas permeability, and f is the fracture spatial frequency. Taking the logarithm of this expression gives:

$$\log(2b) = \frac{1}{3}\log(12) + \frac{1}{3}\log(k_G) - \frac{1}{3}\log(f) \qquad \text{(Eq. 26)}$$

The mean and standard deviation for log(k_G) and f are given in Table 4 for some of the model layers. Taking log(k_G) and log(f) to be normally distributed with means $m_{\log k}$ and $m_{\log f}$, respectively and variances, $\sigma^2_{\log k}$ and $\sigma^2_{\log f}$, respectively, then the mean and variance for log($2b$) are:

$$\mu_{\log 2b} = \frac{1}{3}\log(12) + \frac{1}{3}\mu_{\log k} - \frac{1}{3}\mu_{\log f} \qquad \text{(Eq. 27)}$$

$$\sigma^2_{\log 2b} = \frac{1}{9}\sigma^2_{\log k} + \frac{1}{9}\sigma^2_{\log f} \qquad \text{(Eq. 28)}$$

Table 3: Geometric Mean Fracture Aperture and Spacing

Model Layer	Fracture Frequency (1/m)	Fracture Spacing (m)	Fracture Porosity (dimension-less)	Fracture/Matrix Connection Area (m^2/m^3)	Mean Fracture Aperture (m)
tcw11	0.92	1.09	2.8E-2	1.56	1.79E-02
tcw12	1.91	0.524	2.0E-2	13.39	1.49E-03
tcw13	2.79	0.358	1.5E-2	3.77	3.98E-03
ptn21	0.67	1.49	1.1E-2	1	1.10E-02
ptn22	0.46	2.17	1.2E-2	1.41	8.51E-03
ptn23	0.57	1.75	2.5E-3	1.75	1.43E-03
ptn24	0.46	2.17	1.2E-2	0.34	3.53E-02
ptn25	0.52	1.92	6.2E-3	1.09	5.69E-03
ptn26	0.97	1.03	3.6E-3	3.56	1.01E-03
tsw31	2.17	0.461	5.5E-3	3.86	1.42E-03
tsw32	1.12	0.893	9.5E-3	3.21	2.96E-03
tsw33	0.81	1.23	6.6E-3	4.44	1.49E-03
tsw34	4.32	0.231	1.0E-2	13.54	7.39E-04
tsw35	3.16	0.316	1.1E-2	9.68	1.14E-03
tsw36	4.02	0.249	1.5E-2	12.31	1.22E-03
tsw37	4.02	0.249	1.5E-2	12.31	1.22E-03
tsw38	4.36	0.229	1.2E-2	13.34	9.00E-04
tsw39	0.96	1.04	4.6E-3	2.95	1.56E-03
ch1z	0.04	25	1.7E-4	0.11	1.55E-03
ch1v	0.10	10	6.9E-4	0.3	2.30E-03
ch2v	0.14	7.14	8.9E-4	0.43	2.07E-03
ch3v	0.14	7.14	8.9E-4	0.43	2.07E-03
ch4v	0.14	7.14	8.9E-4	0.43	2.07E-03
ch5v	0.14	7.14	8.9E-4	0.43	2.07E-03
ch2z	0.14	7.14	4.3E-4	0.43	1.00E-03
ch3z	0.14	7.14	4.3E-4	0.43	1.00E-03
ch4z	0.14	7.14	4.3E-4	0.43	1.00E-03
ch5z	0.14	7.14	4.3E-4	0.43	1.00E-03
ch6	0.04	25	1.7E-4	0.11	1.55E-03
pp4	0.14	7.14	4.3E-4	0.43	1.00E-03
pp3	0.20	5	1.1E-3	0.61	1.80E-03
pp2	0.20	5	1.1E-3	0.61	1.80E-03
pp1	0.14	7.14	4.3E-4	0.43	1.00E-03
bf3	0.20	5	1.1E-3	0.61	1.80E-03
bf2	0.14	7.14	4.3E-4	0.43	1.00E-03
tr3	0.20	5	1.1E-3	0.61	1.80E-03
tr2	0.14	7.14	4.3E-4	0.43	1.00E-03
tcwf - faults	1.90	0.526	4.4E-2	1.3E+1	3.38E-03
ptnf - faults	0.54	1.85	1.6E-2	1.3E+0	1.23E-02
tswf - faults	1.70	0.588	3.6E-2	8.6E+0	4.19E-03
chnf – faults & below	0.13	7.69	1.6E-3	4.7E-1	3.40E-03

DTN: LB990501233129.001

In fact, the only expression needed is the variance because the values in Table 4 will be used for the geometric mean fracture aperture. To use the relationship for $\sigma^2_{\log 2b}$, the relationship between the arithmetic mean and standard deviation for *f*, given in Table 4, to the variance for log (*f*) is desired. The relationship between σ^2_f, μ^2_f, and $\sigma^2_{\log f}$ is the following (Iman and Shortencarier 1984, p. 17):

$$\frac{\sigma^2_f}{\mu^2_f} = \exp(A^2 \sigma^2_{\log f}) - 1 \qquad \text{(Eq. 29)}$$

where A= ln(10) is used to convert between logarithms of base 10 to base *e*. Solving for $\sigma^2_{\log f}$ gives

$$\sigma^2_{\log f} = \left(\frac{1}{\ln(10)} \right)^2 \ln\left(1 + \frac{\sigma^2_f}{\mu^2_f} \right) \qquad \text{(Eq. 30)}$$

The results are summarized in Table 4.

Table 4. Aperture Data and Derived Geometric Standard Deviation of Fracture Aperture

Model layer	$m_{\log k}$	$\sigma_{\log k}$	μ_f	σ_f	$\sigma^2_{\log f}$	$\sigma^2_{\log k}$	$\sigma^2_{\log 2b}$	Geometric Std. Dev. of 2b
tcw12	-11.279	0.778	1.91	2.09	0.14848	0.60528	0.08375	1.94715
tcw13	-11.344	1.147	2.79	1.43	0.04399	1.31560	0.15106	2.44722
ptn21	-11.491	0.885	0.67	0.92	0.19987	0.78322	0.10923	2.14044
ptn25	-12.784	0.101	0.52	0.6	0.15965	0.01020	0.01887	1.37207
tsw32	-12.146	0.658	1.12	1.09	0.12568	0.43296	0.06207	1.77477
tsw33	-12.112	0.612	0.81	1.03	0.18144	0.37454	0.06177	1.77235
tsw34	-12.474	0.546	4.32	3.42	0.09177	0.29811	0.04332	1.61486

DTN: LB990501233129.001 (aperture data)

The average geometric standard deviation for these model layers is 1.9. This average geometric standard deviation is to be used for all the model layers beneath the potential repository to perform stochastic sampling of the fracture aperture in the transport calculations for TSPA.

6.2.2 Multiple Radionuclides With Decay/Ingrowth

The FEHM (Zyvoloski et al. 1997) code allows particles to decay with or without the production of the daughter product. For multiple species with decay chains, the code uses the approach outlined below to determine the number of decayed particles, and the code performs the bookkeeping needed to keep track of the locations and numbers of each type of radionuclide. These multiple species can each have their own transport parameters such as sorption and diffusion coefficients.

For decay-ingrowth simulations with time-dependent release of tracer particles, the computational burden increases dramatically with the number of particles in the field. For example, the decay-ingrowth calculation for species *i* decays into species *j* at a decay rate λ is:

$$N_j = \sum_{m=1}^{N_i} \{1 - \exp[-\lambda(t - t_m)]\} \qquad \text{(Eq. 31)}$$

where N_j is the number of particles of species *j* decayed from species *i*, N_i is the number of particles of species *i,* and t_m is the time at which the *m th* particle is injected into the system. If 500,000 particles of species *i* are injected into the system, then at each time step, the number of mathematical operations for ingrowth calculations alone are around 2.5 million. For a simulation time period of 1 million years, the typical calculation requires about 100 time steps. Therefore, the total number of operations for ingrowth calculations will reach 0.25 billion. Therefore, for site-scale simulations, the use of Equation 31 would be extremely inefficient.

To reduce the computational burden in simulations, the decay-ingrowth calculation in Equation 31 is approximated with an integral by assuming that particles are injected into the system uniformly in time domain. Multiplying both sides of Equation 31 by *Δt,* the average injection time interval between particles, and approximate Equation 31 with an integral:

$$N_j \approx \left\{(\tau_1 - \tau_2) + \frac{1}{\lambda} \bullet [\exp(-\lambda\tau_1) - \exp(-\lambda\tau_2)]\right\} \Big/ \Delta t \qquad \text{(Eq. 32)}$$

where $\tau_1 = t - t_1$ and $\tau_2 = t - t_{Ni}$, t_1 is the time at which particle injection starts, t_{Ni} is the time of the Nth injected particle, and *t* is the time at which the decay-ingrowth calculation is carried out. The use of Equation 32 reduces the number of operations within one time step from millions of operations to just 10, which greatly increases the speed of simulations. Validity of this approach is demonstrated in Section 6.3.

The decay-ingrowth model flow chart as implemented in FEHM is summarized below in Figure 4.

Inside FEHM, mass is passed to the particle-tracking module either as direct input or by the Graphical Simulation Environment software package (GoldSim, Golder 2000). When the input is as mass, conversion factors (# of particles/mole) are introduced to convert mass into number of particles for use in particle-tracking simulations. The conversion factors are stored in memory and are passed to the daughter species when parent particles decay into daughter particles. The conversion factors guarantee that mass is conserved during the decay-ingrowth process. At the end of each simulation, FEHM uses the conversion factors to convert number of particles back into mass, then passes the results back to RIP or writes them to an output file.

The accuracy of the integral approach depends on the number of particles and their release history. In general, the use of a greater number of particles increases the accuracy. When the same number of particles is used in simulations, the one with a constant release rate will cause less error. If the release rate changes with time, the release period is divided into segments so that within each segment (corresponds to each time step) the release rate can be treated as a constant.

Additional bookkeeping is required to handle the case of multiple species. For each species, particles released during a time step are stored in the corresponding particle array and are labeled as one segment of the entire array. The particle starting and ending release times, release time interval, and particle indexing information are stored in memory for use in decay-ingrowth calculations. Thus, a particle array consists of many segments with each segment corresponding to particles released at the corresponding time step.

At each time step, particles are uniformly distributed in the field in two ways:

1. When the number of injected particles, *np*, is larger than the number of releasing nodes, *n*, in the field, each releasing node is assigned *m=int(np/n)* particles. Then, the code loops through all release nodes and at each node, evenly releases *m* particles into the field during the current time step. The remaining particles, if *np* is not a multiple of *n*, are injected as in 2. below.

2. When the number of particles, *np*, is smaller then the number of releasing nodes, *n*, a uniform random number generator is used to randomly select *np* nodes from a poll of *n* releasing nodes to release particles during the current time step. By doing so, particles are uniformly distributed over the repository region.

In the FEHM decay-ingrowth model, a first in, first out approach is used to select which particles undergo decay. Alternatively, the approach could have been to first calculate the number of decay particles, *nd*, for each segment, then select decay candidates from a group of particles which are sequentially released during the corresponding particle injection period. Then, for the case in which several particles are injected simultaneously, to make sure that the selection of decay particles is uniform in space and representative, a uniform random number generator is used to select *nd* decay particles from the particle array. But this approach would require that at each time step, the selecting process be run to pick out decay particles, which would increase computational burden. Instead, an alternative approach was developed in which particles are first distributed uniformly in the field, after which they are sorted in ascending order in time domain based on their release time. Finally, the array index of particles having the same release time are randomly changed. By doing this, the sequence of particles with the same release time but at different release nodes are mixed in the array. Once the number of decay particles, *nd*, is calculated, the code transforms the first *nd* particles in the parent particle array into daughter species particles. This approach skips the process of selecting decay particles within each segment at each time step, and hence increases the efficiency of the decay model.

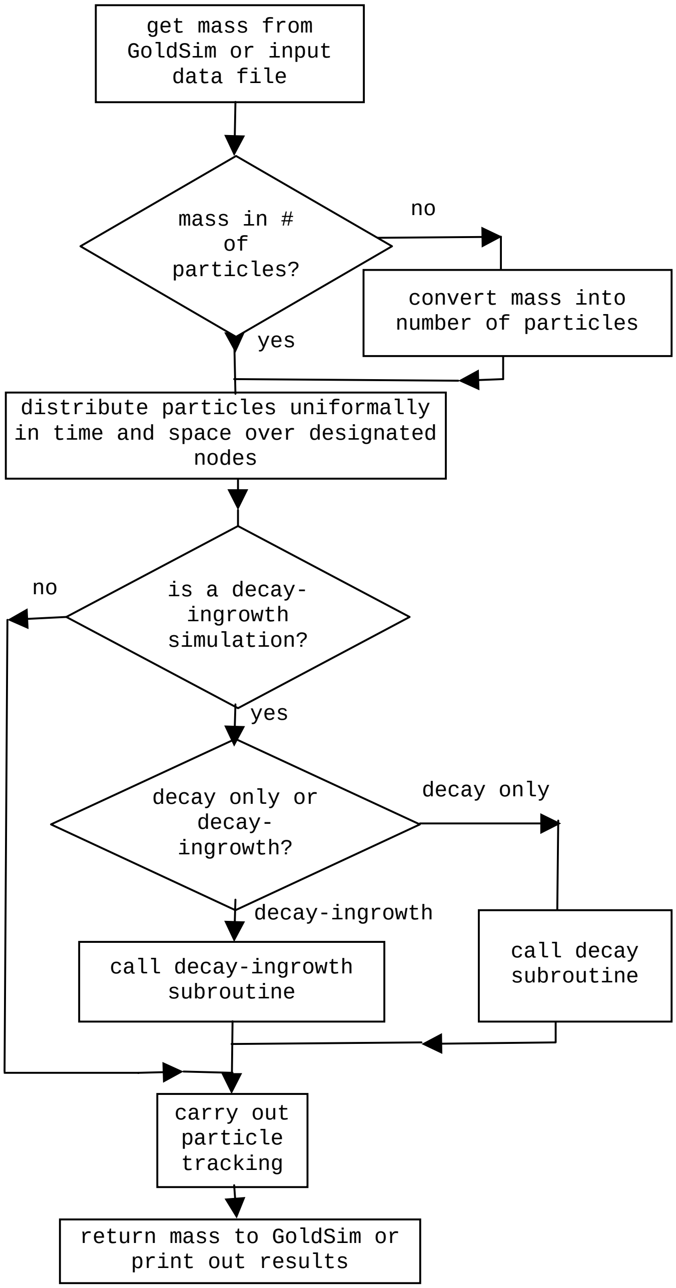

get mass from GoldSim or input data file
mass in # of particles?
no
convert mass into number of particles
yes
distribute particles uniformly in time and space over designated nodes
is a decay-ingrowth simulation?
no
yes
decay only or decay-ingrowth?
decay only
decay-ingrowth
call decay-ingrowth subroutine
call decay subroutine
carry out particle tracking
return mass to GoldSim or print out results

Figure 4. Flow Chart of FEHM Decay-Ingrowth Model

For simulations involving decay without tracking of the daughter species, an indicator is set so that the code skips the decayed particle in the particle-tracking simulations. Figure 5 shows the implementation of decay simulations at each time step in FEHM.

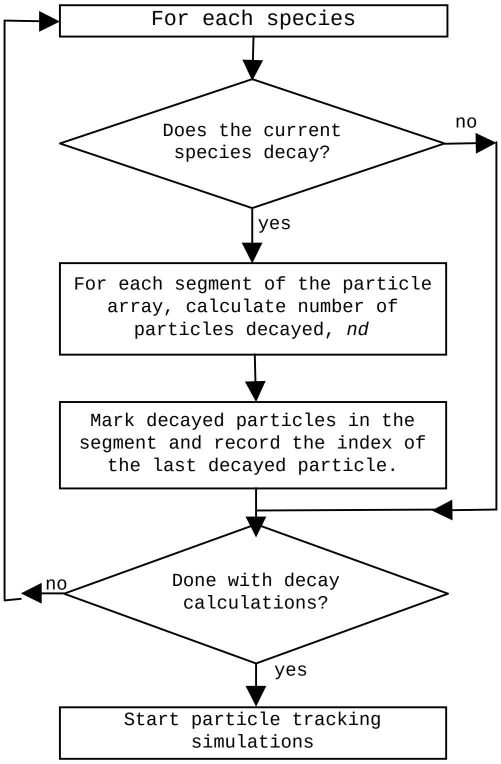

For each species
Does the current species decay?
no
yes
For each segment of the particle array, calculate number of particles decayed, *nd*
Mark decayed particles in the segment and record the index of the last decayed particle.
Done with decay calculations?
no
yes
Start particle tracking simulations

Figure 5. Flow Chart of Decay Subroutine

At each time step, for a decay species, the code loops through each segment and calculates the number of parent particles decayed, *nd*, within the current time step using the integration approach in Equation 32 as follows. Suppose the particle decay index which records the position of the last decayed particle during previous time step for the same segment is *m*. Then, at the current time step, the code marks *nd* particles sequentially in the particle array as decayed by pointing the decay index pointer at (*m*+*nd*) in the array of the parent species. The updated decay index pointers for each segment are stored in the memory for use in next time step. Once the decay calculations are performed, the particle-tracking module is started to simulate the movement of particles in the field. Using the decay index pointers stored in the memory, the code skips particles already decayed previously, thereby reducimg unnecessary operations and CPU time.

Decay calculation for the decay-ingrowth model is the same as for the decay only model, except that after a particle decays, it is transformed into a daughter species particle. The flow chart for the decay-ingrowth simulations at each time step is shown in Figure 6.

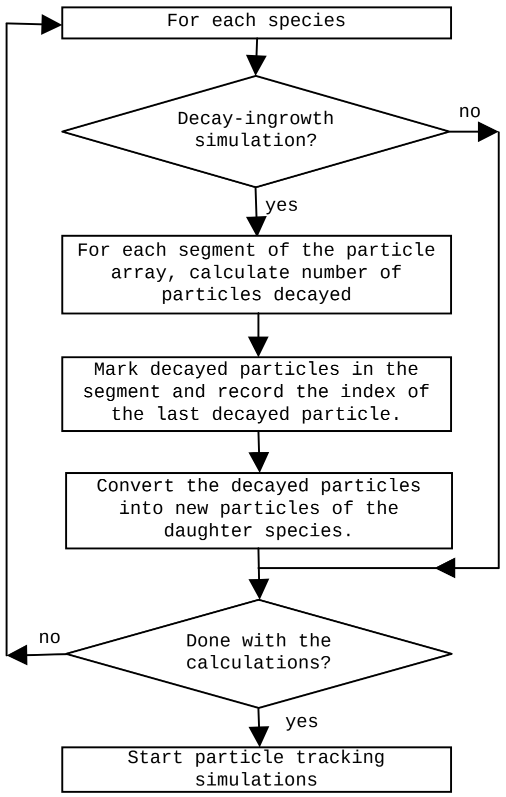

For each species
Decay-ingrowth simulation?
no
yes
For each segment of the particle array, calculate number of particles decayed
Mark decayed particles in the segment and record the index of the last decayed particle.
Convert the decayed particles into new particles of the daughter species.
Done with the calculations?
no
yes
Start particle tracking simulations

Figure 6. Flow Chart of the Decay-Ingrowth Subroutine

A new particle generated due to decay-ingrowth takes over the location of the parent particle but possesses its own transport properties. This process requires that related particle information be passed from the parent particle to the daughter particle. This information includes: particle injection time, particle current residence time in the cell, and the corresponding particle conversion factor, etc. Newly generated daughter particles form a new segment in the daughter particle array. The corresponding starting and ending particle release times for the new segment is stored in the memory for use in sequential decay-ingrowth calculations.

6.2.3 Interface with TOUGH2 and GoldSim

The UZ transport model can be run either as a standalone FEHM simulation or within a performance assessment simulation using the GoldSim application. Although FEHM can perform flow simulations as well, the UZ transport application utilizes only the particle-tracking portion of the code, and thus requires the flow field as input. Flow fields are being determined as part of the UZ PMR using the computer code TOUGH2 (Pruess 1991). The results are converted to FEHM input files using the computer code T2FEHM2 (CRWMS M&O 2000d). The information required to read the flow fields are the grid and connectivity information, the fluid saturation at each grid point, and the fluid mass flux between each node (and any fluid sources and sinks). This conversion program produces input files that can be read by FEHM in lieu of FEHM computing the flow field.

The interface between GoldSim and FEHM establishes a protocol for defining the radionuclide sources to the UZ transport model (provided by GoldSim), the definition of a particular flow field for FEHM to use, and exit mass fluxes of radionuclides from the UZ model (from FEHM to GoldSim based on the particle-tracking simulation). This protocol is quite flexible, allowing an arbitrary number of source regions, radionuclides, exit regions, and flow fields to be defined and passed between GoldSim and FEHM through the FEHM subroutine call statement. During a GoldSim simulation, FEHM cedes control of the time step to GoldSim. At each time, GoldSim provides a flag defining the flow field and the mass flux inputs of radionuclides. By changing the flow field during a simulation, the model can simulate the impact of a climate-induced change in the UZ system. When this occurs, FEHM reads in the new flow field and proceeds with the transport simulation. Since each flow field is a steady state flow field, the implicit assumption is a quasi-steady one, that is, the system establishes a new steady rapidly in comparison to transport velocities through the unsaturated zone. At the end of each time step, FEHM passes back to GoldSim the exiting mass flux values from the UZ model.

6.2.4 Engineered Barrier System (EBS) Random Release Model for Radionuclide Source Terms

If waste packages containing high level radionuclide material in the repository eventually fail due to corrosion, the process will almost certainly be variable in both space and time. At early times, a few packages may fail, releasing radionuclides into the UZ. At later times, a greater number of packages may fail. In VA (Viability Assessment) calculations (DOE 1998, Figure 4-10), releases from the EBS to UZ (Unsaturated Zone) were spread over the entire repository sub-regions. Such treatment of the EBS release could result in significant artificial dilution of the UZ transport source term in some circumstances. In reality, waste packages may not fail uniformly in space and time. Rather, a few waste packages may fail at early times, while others may fail gradually over longer time periods. An EBS random release model was developed in FEHM to allow the model to simulate early failed packages and time- and spatially variable radionuclide releases.

To begin, a repository is defined consisting of N_large sub-regions. Each sub-region contains certain number of waste packages. Initially, M_fine packages fail at locations designated by package x-y locations (x,y). The M_fine failed packages release radionuclides at a mass flow rate of M_flux_i, where i is the ith failed package.

As time proceeds, packages fail in the sub-regions of the repository. At each time step, there are a certain number of failed packages in each sub-region i. The mass flux released from those packages is denoted as N_large_flux for the i^{th} sub-region. In this model, the release nodes in the numerical grid for the failed packages are randomly selected from the available repository nodes within that sub-region to mimic the failure process of the waste packages. The mass release of M_fine packages is separated from those of the other failed packages.

To simulate the impact of the EBS random release on system performance at the Yucca Mountain site, the FEHM EBS random release model was developed to perform the following tasks:

- Locate the M_fine early failed package nodes in repository sub-regions based on given failed package coordinates. If no node matches a given coordinate, then select the nearest node to the given coordinate.

- Randomly select the failed package nodes in the designated sub-region i.

The existing FEHM subroutine *getrip* was modified to handle EBS random release. From FEHM particle-tracking subroutine *part_track*, subroutine *getrip* is called to determine the particle release locations. First, the subroutine obtains information passed by GoldSim in an input array called in[ ]. The structure of the in[ ] array is shown in Figure 7.

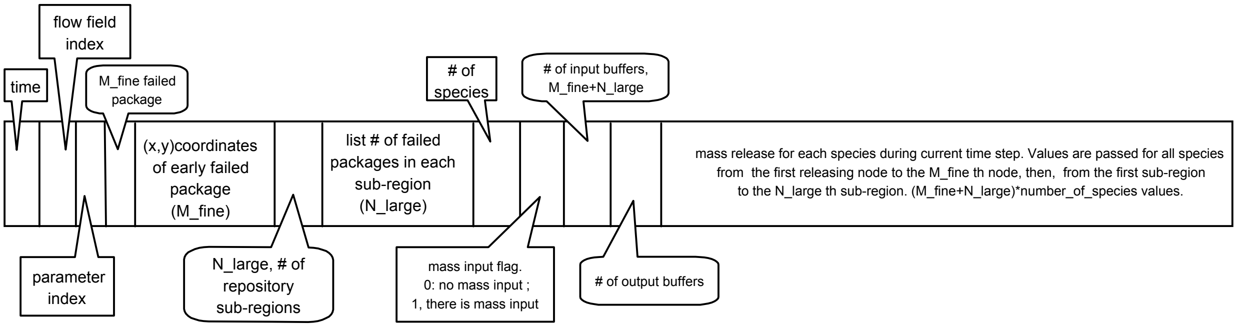

flow field index
time
M_fine failed package
# of species
# of input buffers, M_fine+N_large
(x,y)coordinates of early failed package (M_fine)
list # of failed packages in each sub-region (N_large)
mass release for each species during current time step. Values are passed for all species from the first releasing node to the M_fine th node, then, from the first sub-region to the N_large th sub-region. (M_fine+N_large)*number_of_species values.
parameter index
N_large, # of repository sub-regions
mass input flag. 0: no mass input ; 1, there is mass input
# of output buffers

Figure 7. The Structure of the in[ ] Array Passed to FEHM from GoldSim

The algorithm used in FEHM EBS random release model is summarized in Figure 8, the flow chart of the EBS random release model.

Starting with the M_fine early failed packages, *getrip* extracts the (x,y) coordinates of the early failed packages and loops through each repository sub-region node to select the one that is closest to the given coordinates. To prevent a node being selected more than once for two or more given coordinates, getrip checks the selected nodes for overlapping. If overlapping is found, *getrip* prints out error messages to the error file fehmn.err, then stops the program.

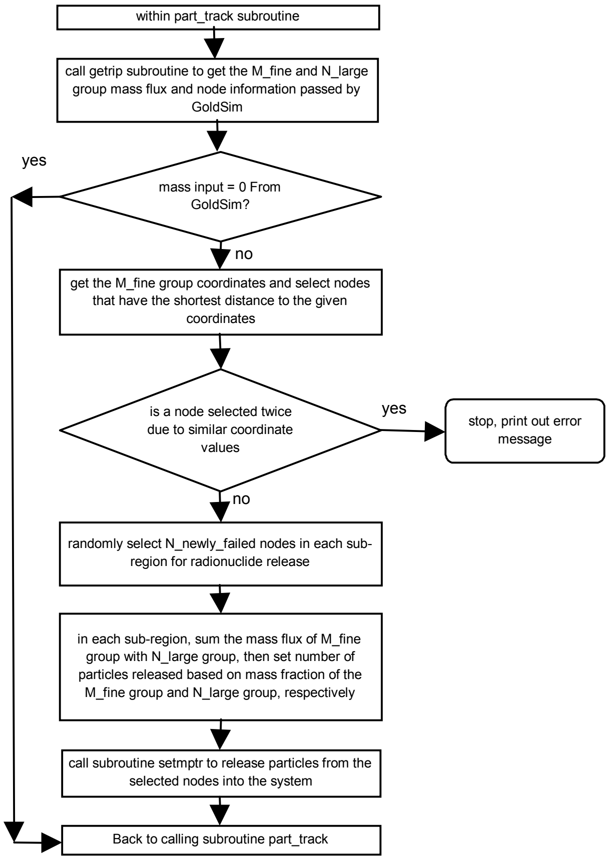

within part_track subroutine
call getrip subroutine to get the M_fine and N_large group mass flux and node information passed by GoldSim
yes
mass input = 0 From GoldSim?
no
get the M_fine group coordinates and select nodes that have the shortest distance to the given coordinates
is a node selected twice due to similar coordinate values
yes
stop, print out error message
no
randomly select N_newly_failed nodes in each sub-region for radionuclide release
in each sub-region, sum the mass flux of M_fine group with N_large group, then set number of particles released based on mass fraction of the M_fine group and N_large group, respectively
call subroutine setmptr to release particles from the selected nodes into the system
Back to calling subroutine part_track

Figure 8. Flow Chart of FEHM Random EBS Random-Release Model

When the selection of M_fine nodes is complete, *getrip* extracts the number of failed packages in each sub-region for the N_large sub-regions. The number of failed packages at the current time step is compared with the values at the previous time step to determine the number of newly failed packages, N_newly_failed, within the current time step in each sub-region. Then, *getrip* randomly selects N_newly_failed nodes within the corresponding sub-region. The selected failed nodes are stored in the memory for use in releasing radionuclides and are removed from the base of available repository nodes. If the number of failed packages is larger than the number of nodes in a sub-region, then radionuclide will be released from all nodes within the sub-region. Once all nodes of failed packages are determined, *getrip* allocates the number of released particles proportionally to the mass flux values of each failed package. Then, subroutine *setmptr* is called to inject particles into the system for each species.

To ensure that the code is functioning as designed, a verification example is included in a subsequent section.

6.2.5 Colloid Transport Parameters for the UZ Transport Model

To simulate colloid size exclusion between the fracture and matrix and filtration processes at the interfaces between matrix unit, arguments are invoked related to the colloid size versus the pore size to determine conservative parameters. The pore size distribution for hydrogeologic units considered in the model can be estimated from moisture retention curves (DTN: GS980908312242.039, GS950608312231.008, GS960808312231.003). The pore sizes were calculated from Marshall et al. (1996, Equation 8.1a), which is

$$r = -\frac{2\gamma \cos\theta}{\psi} \qquad \text{(Eq. 33)}$$

where r is the radius of pores, γ is surface tension of the water, θ is contact angle (assumed to be zero), and ψ is matric potential. This equation is valid for the range of pore sizes between 100 nm and 1000 μm. Many of the pore sizes in units such as the Topopah Springs welded (TSw4) are smaller. A correction for pore sizes less than 100 nm can be calculated if the relative vapor pressure of the soil water is known. This data was not available, so the correction was not made, and a slight error will exist in the pore sizes calculated less than 100 nm. To determine the percent of pores within each size range, the change in moisture content between measurements was used.

The percentage of colloids that can enter a matrix unit from the fractures can be determined based on the percentage of the pores that were greater than the colloid size of interest. For example, for the TSw4, 71% of the pore sizes are estimated to be smaller than 100 nm (DTN: LA0003MCG12213.002). Therefore, for the size exclusion between the fracture and matrix only 29% of 100 nm colloids are allowed to enter the matrix (Table 5). The remainder of the colloids must therefore remain in the fracture continuum. The values used for the size exclusion are based on an expected colloid size of 100 nm, but values based on other colloid sizes could also be used.

Colloid filtration at interfaces between matrix units is based on the colloid size distribution and the cumulative probability of colloids entering a unit based on the pore size distributions (Table 6). The implementation of distributions allows for a wide range of colloid sizes to be considered in a single simulation. For filtration at the interface between two units, the properties of the unit that flow is entering is used. The data in Table 6 show that the only interfaces where colloid filtration is likely to occur is between units like the vitric and zeolitic Calico Hills where there is a large contrast in pore size distribution. When a colloid can not enter a matrix unit at an interface, the colloid is removed from the simulation and is considered permanently filtered. As a result, the use of filtration at interfaces is not recommended for natural colloids associated with actinides because when the colloids are removed they no longer participate in reactions, such as desorption.

These estimates are only based on the pore size distribution, discounting any effect of partial saturation. In the Tsw4 and the Ch1z, the average saturation is close to one, so any variations in water content would not significantly affect the values provided. For the CH1v, the average saturation is much lower. The use of the pore size distribution under fully saturated conditions is justified by recognizing that the saturation is probably higher near the fracture matrix interface where the size exclusion rule is applied and lower further away from the fracture. Therefore, the matrix would probably be sufficiently saturated near the fracture that a reduction in the percent of colloids that can enter the matrix due to unsaturated conditions was not required.

Table 5. Fraction of 100-nm Colloids That Can Enter the Matrix from the Fracture Based on Size

Units	
TMN (TSW4)	0.29
TLL (TSW5)	0.39
TM2 (TSW6)	0.35
TMN1 (TSW7)	0.07
PV3 (TSW8)	0.10
PV2 (TSW9)	0.61
BT1a (CH1)	0.15
CHV	0.61
CHZ	0.27
BT (CH6)	0.08
PP4	0.02
PP3	0.79
PP2	0.35
PP1	0.43
BF3	0.26
BF2	0.04

DTN: LA0003MCG12213.002

To summarize the size exclusion and filtration information, our recommendation is to use colloid-size distribution information to set the colloid transport parameters based on the information in Tables 5 and 6.

To estimate retardation of colloids in the fracture continuum, field experiments at the C-wells complex near Yucca Mountain were examined, in which transport of microspheres was used as an analog for colloids. The tracer experiments were conducted in both the conductive Bullfrog unit and less conductive Prow Pass unit. The microsphere breakthrough curves were then fit to forward and reverse filtration rates (DTN: LA0002PR831231.003). These were then used to calculate a retardation factor for colloid transport through saturated fractured rock (Table 7). Data on colloid filtration through partially saturated fractures is not available. Therefore, it is recommended that either: 1) the distribution of retardation factors for the saturated zone be used to represent colloid retardation through unsaturated fractures, or 2) conservatively assume no colloid retardation in fractures.

Table 6. Cumulative Probabilities for Colloid Transport between One Matrix Unit and Another

Units	Colloid Size (nm)						
	2000	1000	450	200	100	50	6
TMN (TSW4)	1.00	0.92	0.87	0.81	0.71	0.55	0.31
TLL (TSW5)	1.00	0.80	0.79	0.70	0.61	0.51	0.19
TM2 (TSW6)	1.00	0.94	0.90	0.82	0.65	0.51	0.21
TMN1 (TSW7)	1.00	0.99	0.99	0.99	0.93	0.68	0.36
PV3 (TSW8)	1.00	0.98	0.96	0.94	0.90	0.89	0.68
PV2 (TSW9)	1.00	0.72	0.57	0.47	0.39	0.35	0.22
BT1a (CH1)	1.00	0.91	0.89	0.87	0.85	0.83	0.53
CHV	1.00	0.58	0.49	0.43	0.39	0.36	0.07
CHZ	1.00	0.79	0.76	0.73	0.68	0.56	0.30
BT (CH6)	1.00	0.95	0.94	0.92	0.92	0.85	0.40
PP4	1.00	0.99	0.99	0.98	0.98	0.96	0.32
PP3	1.00	0.49	0.34	0.26	0.21	0.16	0.07
PP2	1.00	0.91	0.86	0.81	0.65	0.53	0.22
PP1	1.00	0.79	0.68	0.63	0.57	0.48	0.21
BF3	1.00	0.97	0.94	0.83	0.74	0.66	0.14
BF2	1.00	0.98	0.97	0.96	0.96	0.83	0.25

DTN: LA0003MCG12213.002

Table 7. Cumulative Probability Density Functions for the Retardation of Colloids Through Saturated Fractured Rock

R (retardation factor)	Probability
1.06	0.0105
1.1	0.039
6	0.08125
100	0.2605
280	0.7605
800	1

DTN: LA0002PR831231.003.

6.3 CODE VERIFICATION

In this section, a series of comparisons are presented of the particle-tracking model against analytical solutions or other numerical model results. In all cases, the acceptance criteria for these simulations is based on a visual comparison. The comparison is performed to ensure that the particle-tracking model is reproducing the analytical or other solution result to within the resolution of the plot. The development of more quantitative criteria is difficult because comparison of particle-tracking results with other solutions is subject to considerable random error associated with the number of particles injected. Therefore, scientific judgement in the visual comparison of results is a more appropriate means for evaluating the particle-tracking model.

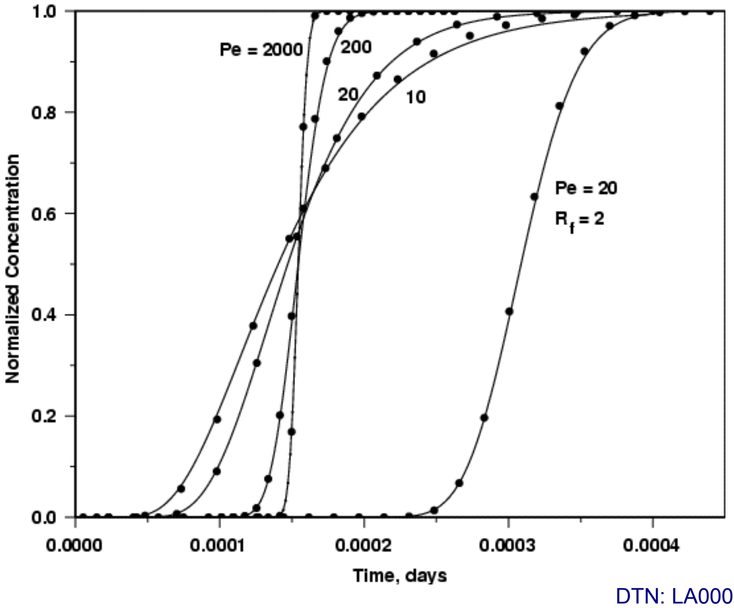
6.3.1 One-Dimensional Transport with Dispersion and Sorption

The first set of simulations are one-dimensional flow problems for which the dispersion and sorption can be tested against analytical solutions. The flow system is a simple one-dimensional grid in the direction of flow. A steady-state flow field is established for a ten-node, one-dimensional grid, after which the RTTF particle-tracking technique is used to simulate the transport.

In the first suite of one-dimensional tests, a short pulse of particles is injected into the system to test the dispersion transfer function. The responses of these pulse injections of solutes are compared to the time-derivative of Equation 7, derived by Nauman and Buffham (1983, p. 107, Equation 3.47):

$$C = \frac{1}{2\theta}\exp\left[\frac{-Pe(1-\theta)^2}{4\theta}\right]\sqrt{\frac{Pe}{\pi\theta}} \qquad \text{(Eq. 34)}$$

The cumulative RTD is also compiled from the particle statistics at the outlet node and compared to Equation 7. Figure 9 shows that the cumulative RTD solution is reproduced almost exactly for $Pe > 10$, corresponding to a grid Peclet number of 1, as long as a sufficient number of particles is used in the simulation. For this transport system, the $Pe = 20$ results become affected by statistical fluctuations when fewer than 10,000 particles are used, whereas for 100,000 particles, the breakthrough curve is represented quite accurately, as shown in Figure 10. This effect is problem dependent; a sufficient number of particles must be determined empirically for each application of the particle-tracking technique by comparing the results with a simulation using more particles.

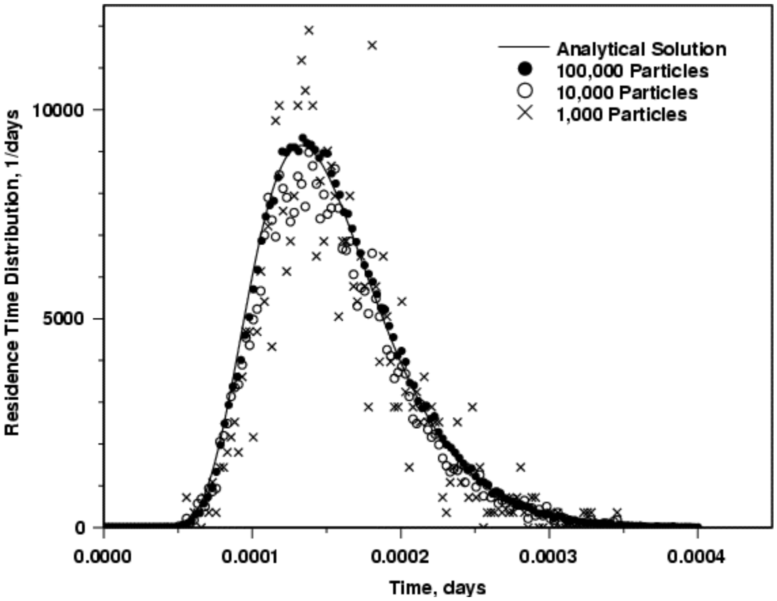


DTN: LA0002BR12213S.002

NOTE: The family of breakthrough curves of mean travel time of 0.00015 days are for R_f = 1.

Figure 9. Cumulative Breakthrough Curves Produced by the RTTF Particle-tracking Technique for One-Dimensional Dispersion (Filled Circles) Compared to Analytical Solution (Solid Curves)

Figure 9 also shows that for low enough values of Pe, such as the $Pe = 10$ case in this example, the RTTF particle-tracking simulation begins to deviate noticeably from the analytical solution. This case corresponds to a grid Peclet number of 1, which is a practical cut-off below which one cannot obtain accurate results using this technique. The solution can be made more accurate using a coarser finite element grid, keeping in mind that the restrictions for obtaining an accurate *fluid flow* solution still exist. The practical implication of this result is that the RTTF particle-tracking technique is most useful for advection-dominated problems (large values of Pe). This will generally be the case for large-scale two- and three-dimensional problems, which necessarily result in large spatial discretization due to computational limitations.



DTN: LA0002BR12213S.002

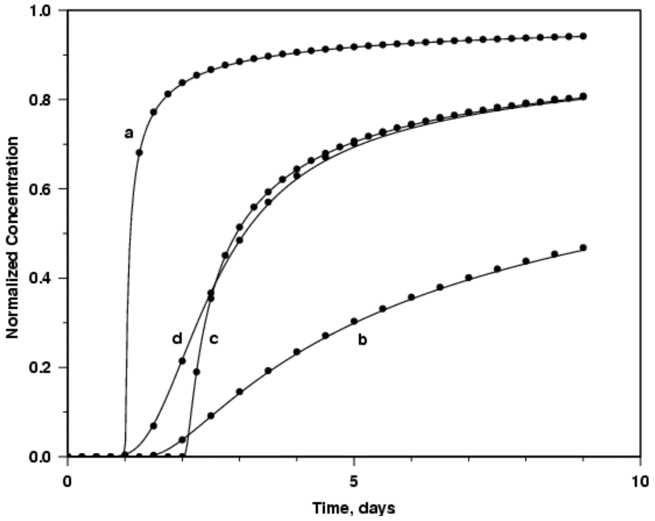
NOTE: The RTD is solute concentration normalized so that the area under the curve is unity. All simulations used Pe=20, R_f=1.

Figure 10. Breakthrough Curves Calculated Using Different Numbers of Particles

Finally, the agreement of the sorption curve in Figure 9 with the analytical solution (Equation 7) shows that linear, equilibrium sorption is also properly accounted for by the RTTF particle-tracking model.

6.3.2 One-Dimensional Transport with Matrix Diffusion

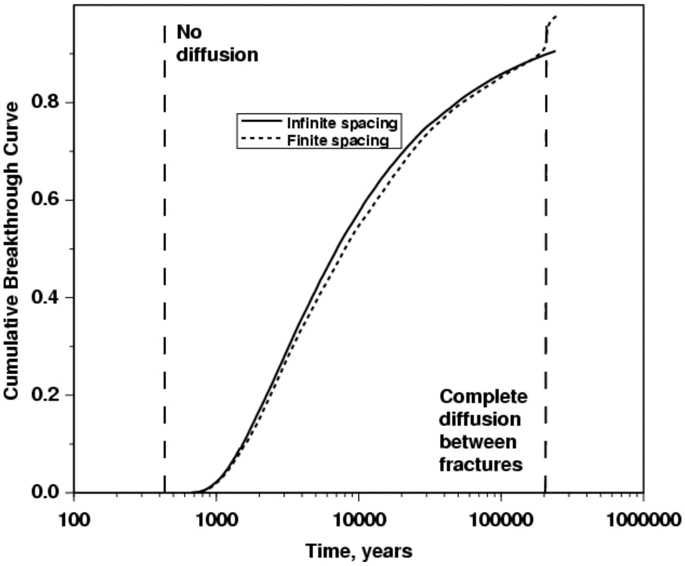
Figure 11 shows a series of simulations of the RTTF particle-tracking algorithm, along with the solution of Tang et al. (1981, Equation 35) for matrix diffusion with and without dispersion in the fractures. The agreement with the analytical solution is very close for both small or moderate amounts of diffusion (curves a and b), for the case of diffusion and sorption on the fracture surface and in the matrix (curve c), and for dispersion, diffusion, and sorption in both media (curve d). This agreement shows that the two-step RTTF approach outlined in the Mathematical Development section provides an excellent approximation of the combined diffusive-dispersive transport system. When the finite-fracture-spacing option in the FEHM particle-tracking model is used, the model (Figure 12) captures the extremes of behavior ranging from no diffusion, to complete diffusion between fractures, to an intermediate case (shown to agree closely with the infinite spacing curve in the figure).



DTN: LA0002BR12213S.002

NOTE: Key to breakthrough curves: a) low matrix diffusion; b) moderate matrix diffusion; c) matrix diffusion and sorption on fracture and in matrix; d) dispersion, matrix diffusion, and sorption on the fracture and matrix.

Figure 11. Breakthrough Curves Computed Using the RTTF Particle-tracking Technique for the Infinite Spacing Matrix Diffusion Model, Compared to the Analytical Solution of Tang et al. (1981) (Solid Curves)



DTN: LA0002BR12213S.002

NOTE: Key to breakthrough curves: Dashed curves - extreme values of no diffusion, and complete diffusion between fractures. Solid curve - infinite fracture spacing assumption. Dotted curve - finite spacing assumption.

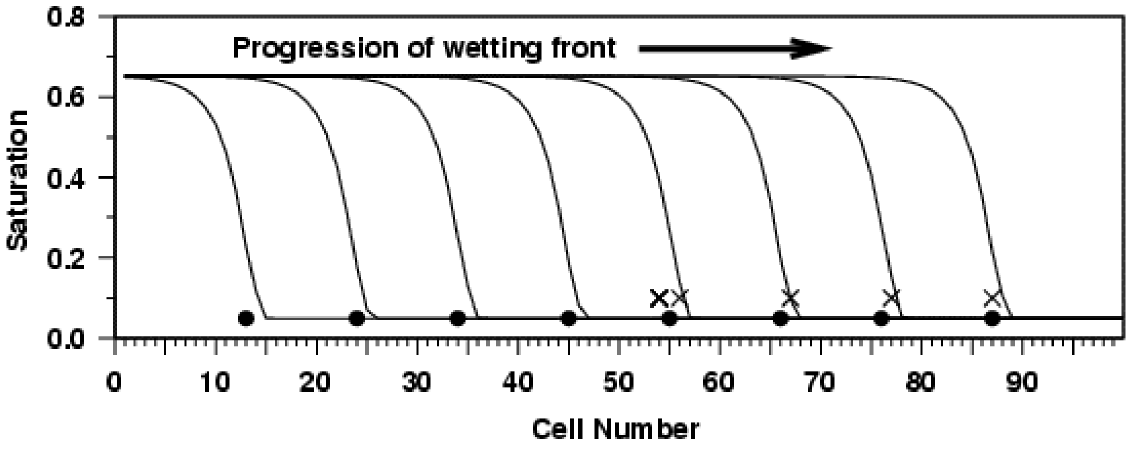
Figure 12. Breakthrough Curve Computed Using the RTTF Particle-tracking Technique for the Finite Spacing Matrix Diffusion Model

In this simulation, a one-dimensional, saturated flow model was used to establish that over a wide range of transport parameters, the code behaves as anticipated. The two dashed curves reproduce the plug-flow breakthrough curves for transport only in the fractures (no diffusion), and complete solute diffusion to the centerline between fractures, resulting in a transport time governed by the matrix porosity. These breakthrough times agree closely with the expected times for fracture-dominated and matrix-dominated transport. At intermediate diffusion coefficients, a comparison of the infinite fracture spacing solution and the finite spacing solution are shown. For the particular parameters shown, the curves agree with each other, indicating that the characteristic diffusion distance is less than fracture spacing for this case. The significant result for this validation exercise is that the two transfer function models agree closely, despite the completely different formulations and governing equations. This is strong evidence that the finite spacing solution is properly implemented in the particle-tracking model.

6.3.3 Transient Wetting of a 1-D Column

In this simulation, a transient, one-dimensional, unsaturated flow system is simulated. Using a relative permeability model proposed by van Genuchten (1980), a model is constructed of the movement of a water front under the influence of gravity and associated tracer particles. The entire flow path is initially set at a saturation below the residual liquid saturation of the medium, and a constant infiltration rate is applied at the inlet of the column. Two particle-tracking simulations are performed: one in which the particles are injected with the infiltrating fluid, and one in which the particles start at a location halfway down the column (cell 54 out of a total of 107). Although many particles are injected, all transport with identical travel times, since this is an advection-only transport case. This validation problem tests the ability of the code to handle the situation of a transient flow field. The exercise is thus important to justify the use of the particle-tracking method for examining radionuclide migration in transient flow fields.

When the particles are injected with the infiltrating fluid, the particles should move with the saturation front. Figure 13 shows that the expected behavior is approximated (the filled circles in the plot), with the particles lagging the front edge of the wetting front by two cells. This slight lag of the particle movement is caused by a slightly dispersed wetting front that does not maintain a sharp interface. For the situation in which the particles are introduced halfway down the length of the column (the X's), the particles remain stationary until the saturation front reaches the particles, after which time the particles move with the front as before. The figure demonstrates that for this transient flow case, the RTTF particle-tracking approach provides an adequate solute transport simulation.



DTN: LA0002BR12213S.002

NOTE: Key to plot. Curves - saturation profiles at regular time intervals. Filled circles - locations of particles started at the entrance. X's - particles started part way down the column.

Figure 13. Movement of Particles with the Saturation Front for the Transient Wetting Case

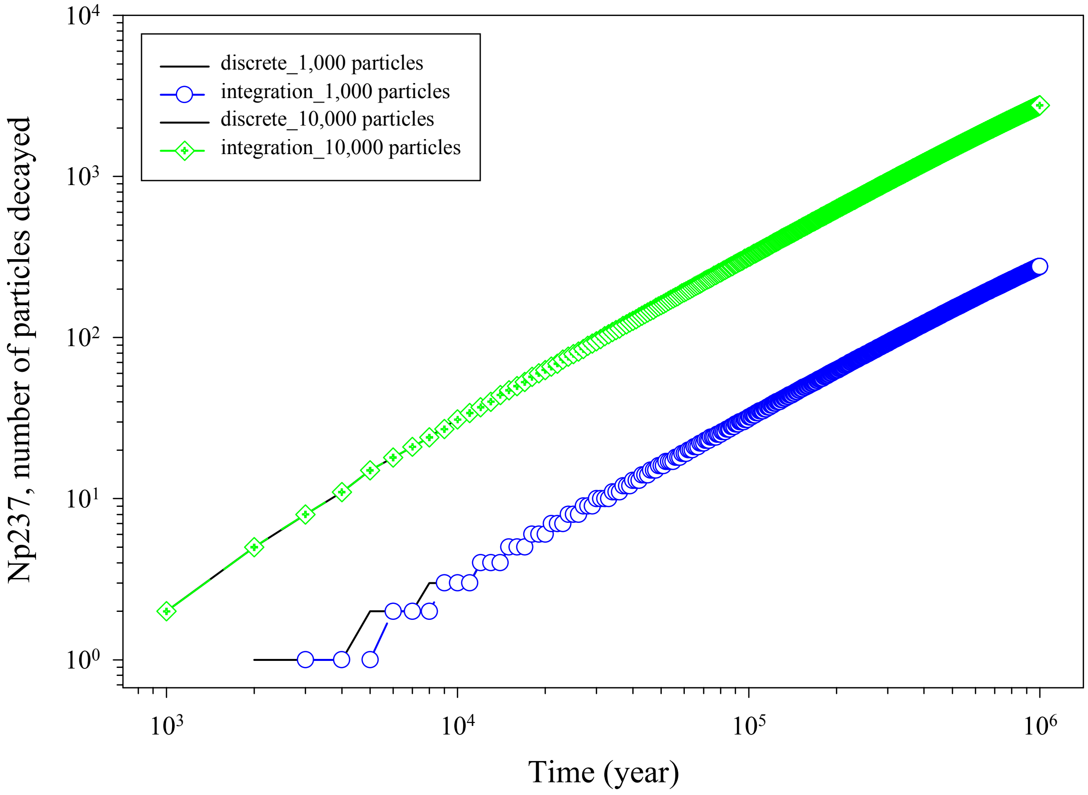
6.3.4 Decay Calculations Using the Integral Approach

A C++ software routine TEST was written to test the integration approach against the discrete formula in Equation 31. The software routine was checked following AP-SI.1Q procedures for correctness. The results showed that routine TEST performed as designed. The source code and the checking process are documented in Attachment I.

Theoretically, the accuracy of the integration approach depends on two factors, the number of particles injected into the system, and the particle injection distribution. In general, large number of particles and uniform injection of particles in time domain increases the accuracy of the integration approach. In a typical performance assessment simulation, on the order of 1,000,000 particles might be released for each species over the repository area (represented by about 500 nodes) during a time period of 10,000 to 1,000,000 years, and the time step used in FEHM simulations is of order 1,000 years. To test the accuracy of the integration approach, the software routine TEST was run by injecting 1,000 particles into the repository during a period of 1,000 years. The results were used to compare the integration approach against the discrete approach.

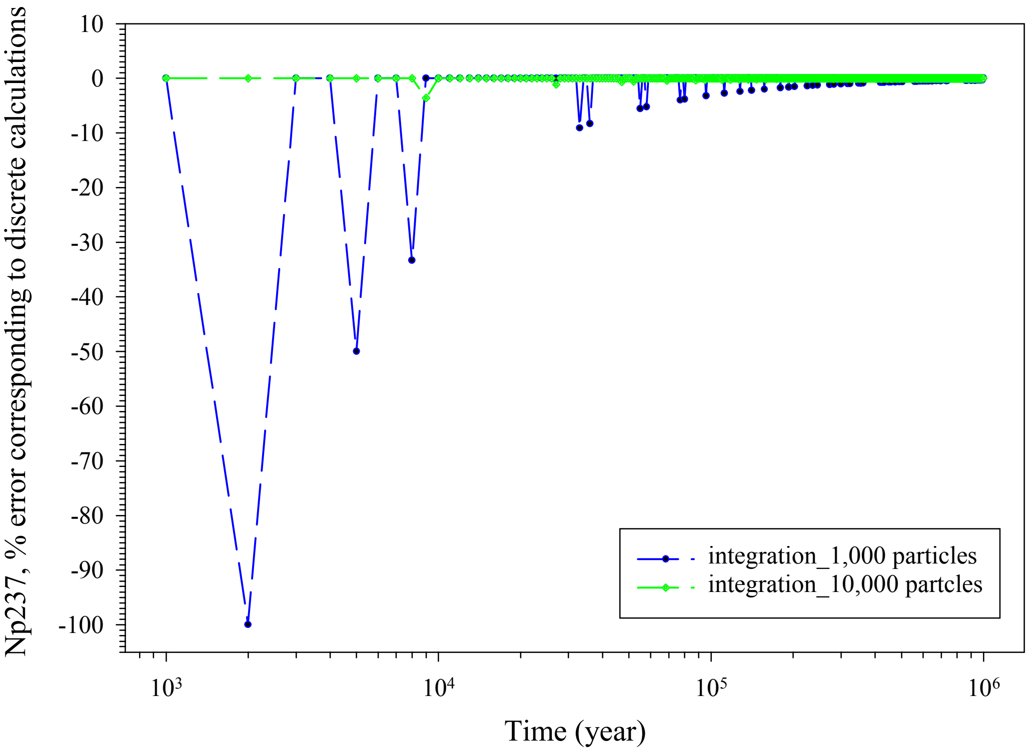
Radionuclides ^{239}Pu (half life of 2.406E4 years) and ^{237}Np (half life of 2.14E6 years) (CRC 1991) were chosen for these tests. Figures 14 and 16 compare the numbers of particles decayed using the two approaches for ^{237}Np and ^{239}Pu, respectively. The results show good agreement between the integration approach and the direct discrete calculation. Figures 15 and 17 are the percentage errors of integration approach corresponding to discrete calculation results at different times. The large relative errors at early times are due to the fact that few particles have decayed. At greater times, the relative percentage error decreases. The relative error can also be interpreted as a time delay of the integration approach corresponding to the discrete calculations in the time domain. The figures also show that increasing the number of particles released during the same time period increases the accuracy of the integration approach.

Therefore, in general, the results from the integral approach are very close to those based on the discrete calculations in Equation 31. The relative error for the majority of the runs was less than 1%. However, the computational burden was greatly reduced from millions of operations to just 10 operations within each time step, and thus is suitable for use in the particle-tracking simulations. Precautions should be taken to ensure that a sufficient number of particles are injected to avoid error. This is best done by increasing the number of particles until the results no longer show significant change from a simulation with fewer particles.



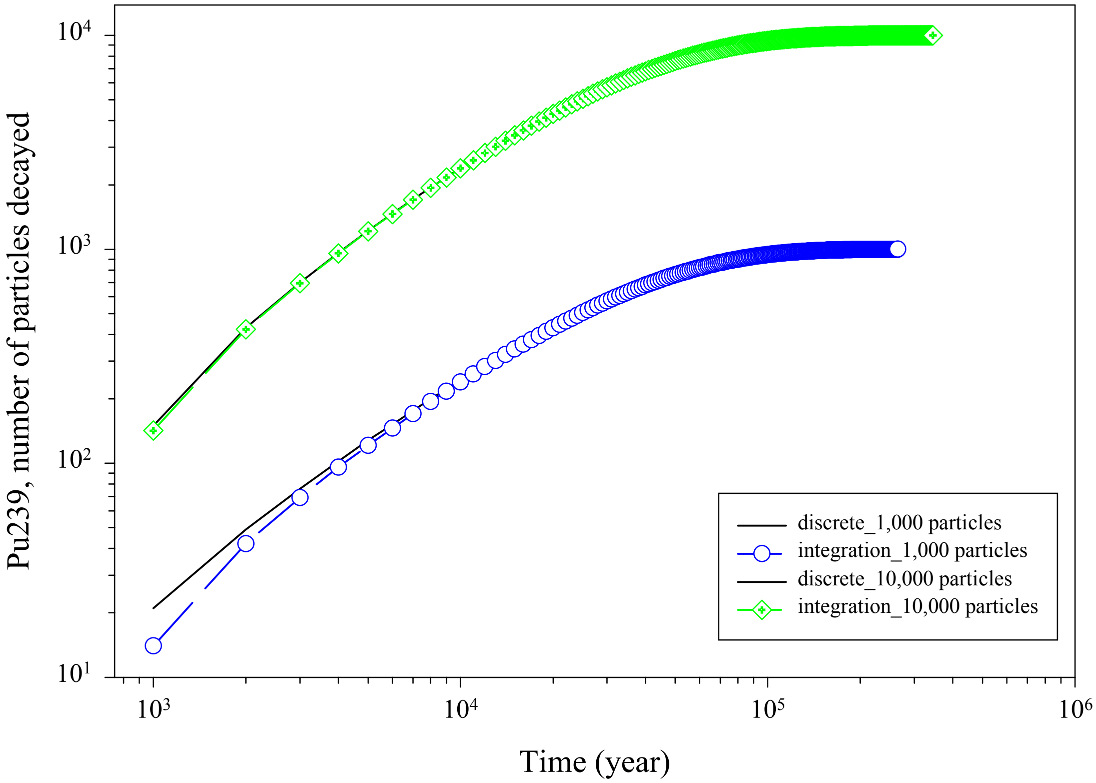
DTN: LA0002BR12213S.002

Figure 14. Number of ^{237}Np Particles Decayed Based on the Integration Approach and the Discrete Calculations



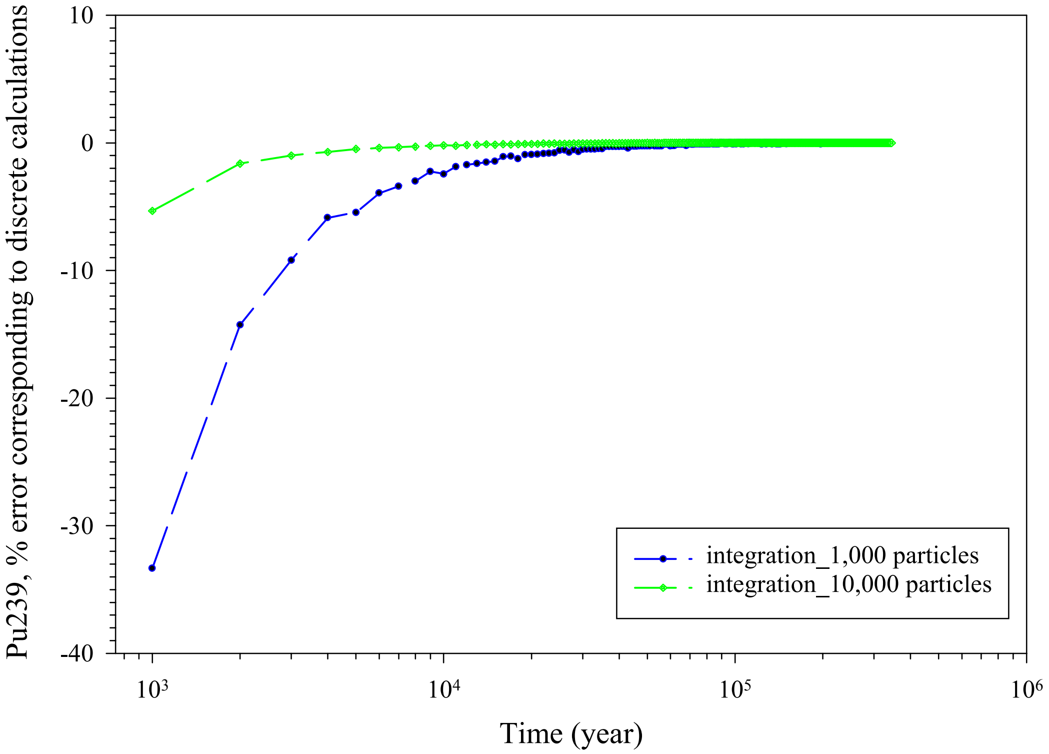
DTN: LA0002BR12213S.002

Figure 15. ^{237}Np Integration Approach Percentage Errors Corresponding to Discrete Calculations



DTN: LA0002BR12213S.002

Figure 16. Number of ^{239}Pu Particles Decayed Based on the Integration Approach and the Discrete Calculations



DTN: LA0002BR12213S.002

Figure 17. ^{239}Pu Integration Approach Percentage Errors Corresponding to Discrete Calculations

6.3.5 Decay/Ingrowth Validation Simulations

In this test suite, simulation results from FEHM particle-tracking model with decay and ingrowth verified against the CHAIN semi-analytical solutions for a 4 species chain decay-ingrowth model (van Genuchten 1985). The use of this routine, the qualification procedures, source code, and test results are listed in Attachment III. The method of comparison for these runs a visual comparison of the plotted results. These runs are designed to test the accuracy of the particle-tracking model, and should be interpreted as representative results of the Yucca Mountain system. The Software routine BKPM was used to post process FEHM simulation results to extract breakthrough information for comparison with CHAIN analytical results. Routine BKPM was qualified according to AP-SI.1Q. The use of the BKPM routine, the qualification procedures, source code, and test results are listed in Attachment II.

For all comparisons of CHAIN and FEHM for decay-ingrowth simulations, a flow system was developed with the following attributes:

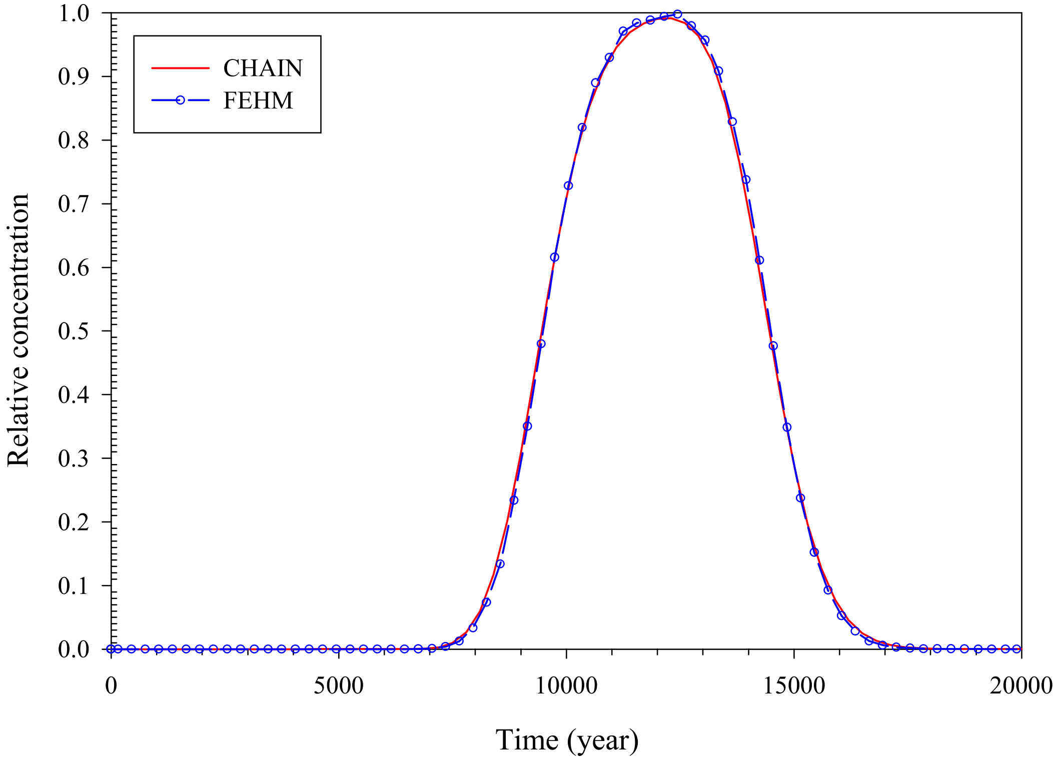
- Saturated steady state flow in a 1-D system
- Porosity of 0.3
- Average pore-water velocity was 1.05192x10^{-4} m/year

- Solute with constant injection concentration of 1.0 mol/l released from 0 to 5,000 years at origin *x*=0. Breakthrough data were collected at *x*=1 m down stream.

- FEHM grid resolution: 0.005 m

- Longitudinal dispersivity of 0.005 m.

The first comparison is for a conservative tracer with no decay and no sorption. The purpose of this initial run is to lay a foundation for FEHM decay-ingrowth model verifications by ensuring that the two codes yield similar transport behavior in the absence of decay and ingrowth. The extracted FEHM particle breakthrough curve is plotted together with the breakthrough curve calculated by CHAIN in Figure 18. The close agreement of the FEHM and CHAIN breakthrough curves proves that the two codes compute the transport of a conservative solute correctly.

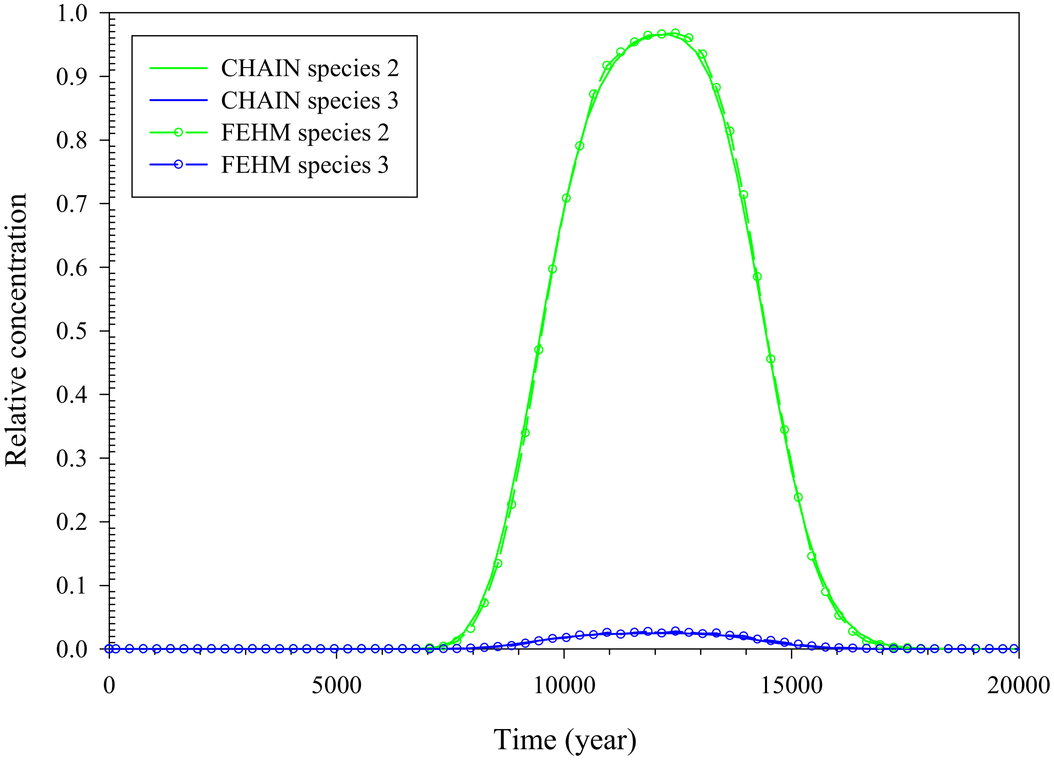


DTN: LA0002BR12213S.002

Figure 18. Comparison of Base Case Transport Results: CHAIN and FEHM

The second test uses the same flow system, but employs a 4 member decay-ingrowth chain. The sequential decay-ingrowth chain was formed by $^{238}Pu \rightarrow ^{234}U \rightarrow ^{230}Th \rightarrow ^{226}Ra$. The decay rates for the 4 species are 0.0079, 0.0000028, 0.0000087, and 0.00043 yr^{-1} for ^{238}Pu, ^{234}U, ^{230}Th, and ^{226}Ra, respectively (van Genuchten 1985, p. 144, appendix 5). The grid Peclet number of the system was 1.0. Retardation factors were set to 1.0 for ^{238}Pu and ^{234}U, but to avoid some parameters being divided by 0 in software routine CHAIN, the retardation factors for ^{230}Th and ^{226}Ra were set to 1.001 in the CHAIN input file. The initial concentrations for all species were 0. From time 0 to 5,000 years, only ^{238}Pu was released into the system at the origin, with a constant concentration of 1.0. In FEHM simulations, the 5,000 year release period was divided into 50 segments so that each segment corresponding to 100 years. Within each segment, 10,000 particles were injected into the system uniformly over the 100 year period. There were no source terms for ^{234}U, ^{230}Th, and ^{226}Ra.

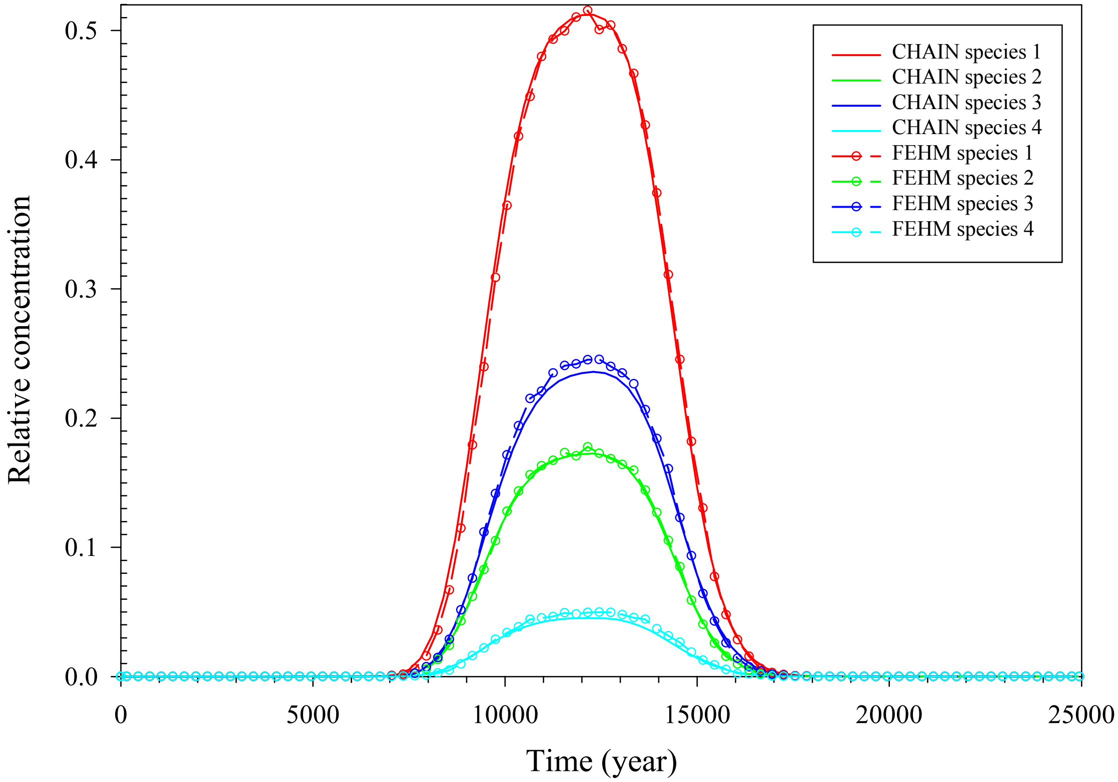
The CHAIN breakthrough data and the post processed FEHM breakthrough data were plotted together in Figure 19. Due to the short half life of ^{238}Pu and the relatively long half lives of ^{234}U and ^{230}Th, the concentrations of ^{238}Pu and ^{226}Ra were extremely low and were omitted in this figure. The figure exhibits good agreement between FEHM and CHAIN breakthrough curves.



DTN: LA0002BR12213S.002

Figure 19. Comparison of CHAIN and FEHM Transport Results for a 4-Member Decay-Ingrowth Chain, $^{238}Pu \rightarrow {}^{234}U \rightarrow {}^{230}Th \rightarrow {}^{226}Ra$. Peclet Number = 1.0

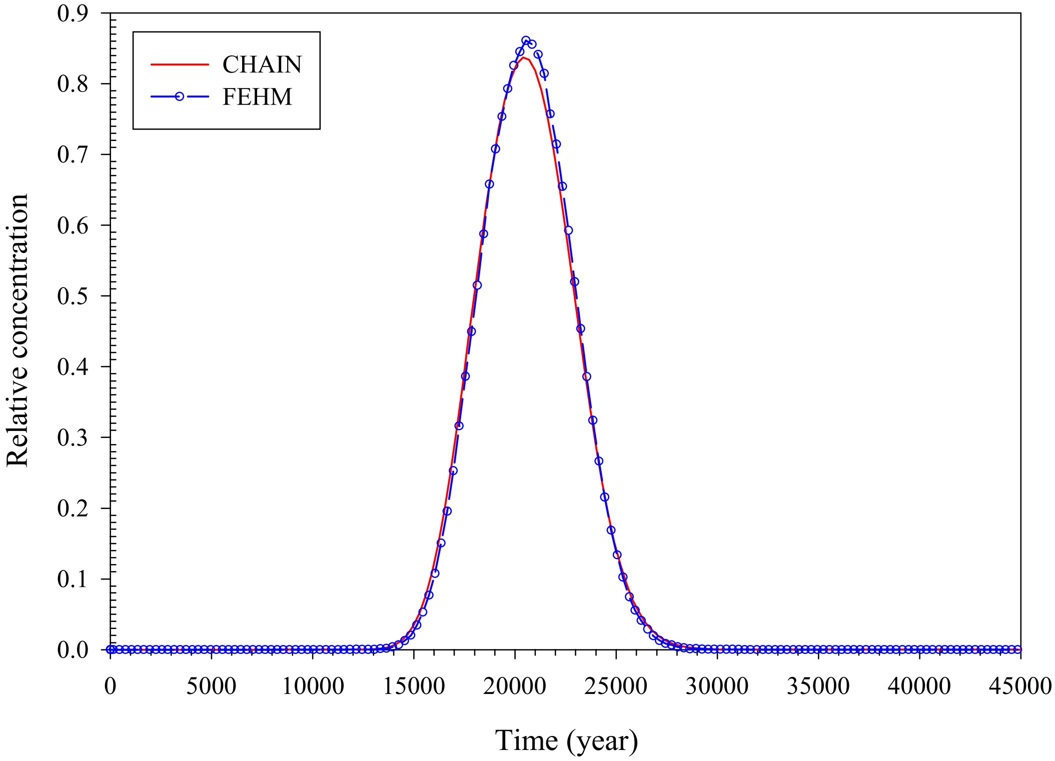
In the third test case, a pseudo sequential decay chain with species $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, with half lives for species 1 through species 4 of 10,000, 3,000, 10,000, and 4,000 years, respectively. The transport process was dominated by advection and dispersion only with a grid Peclet number of 1.0. The particle injection pattern used in this simulation was the same as for case 1 and case 2. The purpose of this problem is to select half-lives that allow all species to exhibit a visible breakthrough at the exit, thus allowing a more complete comparison to be made. The CHAIN and FEHM breakthrough curves are compared in Figure 20. All species agree fairly closely with one another, with only small differences for species 3 and 4, especially near the peak. The differences are probably due to either numerical truncation errors in CHAIN for approximation errors in FEHM. Nevertheless, the visual comparison yields no systematic differences, so the agreement is satisfactory.



DTN: LA0002BR12213S.002

Figure 20. Comparison of CHAIN and FEHM Transport Results for a Case of 4-Member Sequential Decay Chain without Retardation. Peclet Number = 1.0

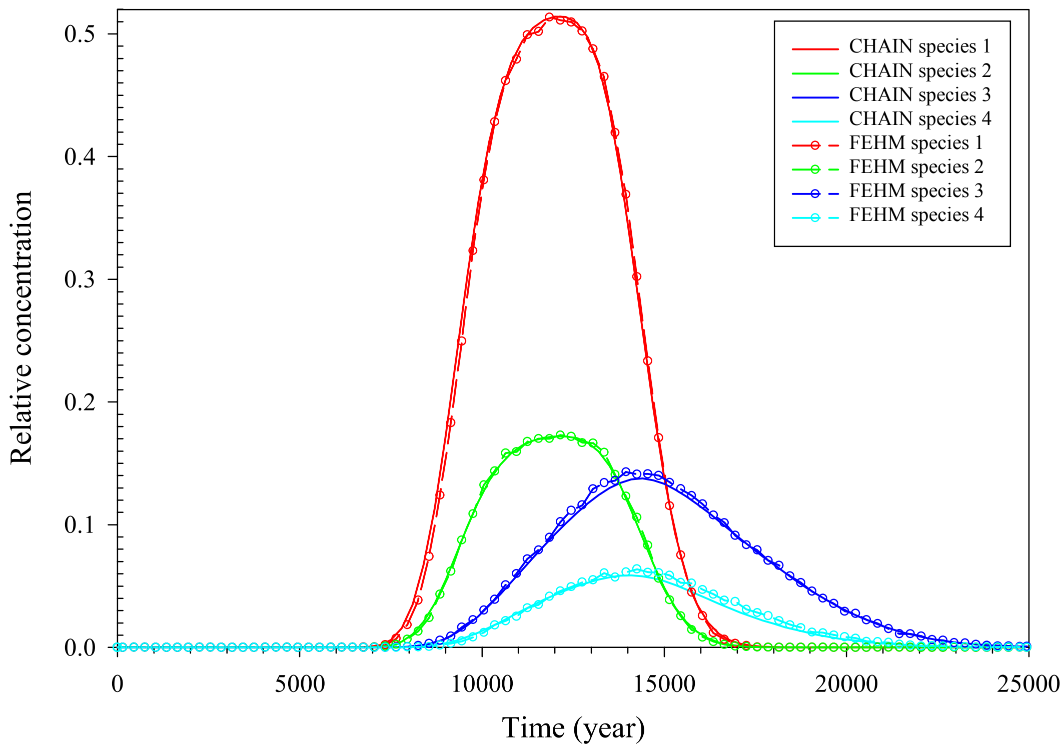
The fourth test case compares FEHM and CHAIN for a single solute with a retardation factor of 1.9. All other parameters are similar to the previous test cases. The pupose of this run is to ensure that the codes are handling sorbing solutes in the same manner as before, including decay and sorption. The comparison shown in Figure 21 shows that this is the case.



DTN: LA0002BR12213S.002

Figure 21. Comparison of CHAIN and FEHM Transport Results for the Base Case with a Retardation Factor of 1.9 and a Peclet Number of 1.0

The fifth and final test case of this suite includes sorption and decay. The retardation factors for species 1 through species 4 were 1.0, 1.0, 1.9, and 1.001, respectively. The boundary and initial conditions and half-lives were the same as those of case 3. The FEHM particle injection pattern was the same as for case 3. The FEHM and CHAIN breakthrough curves were plotted in Figure 22. In general, good agreements were observed between FEHM and CHAIN curves. Taken together, this suite of verification simulations demonstrates that the particle-tracking model accurately handles decay chains and, thus, is suitable for use in simulating multiple radionuclides in the UZ transport model.



DTN: LA0002BR12213S.002

Figure 22. Comparison of CHAIN and FEHM Transport Results for a Case with a 4-Member Decay-Ingrowth Chain and a Retardation Factor of 1.9 for Species 3. Peclet Number = 1.0

6.3.6 EBS Random Release Model Verification

The FEHM EBS random release model was tested in standalone mode to verify that the model worked as designed. The model was tested with a "pseudo problem" to make sure the model selects the right nodes. The test results from the test problem were checked manually for correctness. The EBS random release model was tested for selecting early failed package nodes based on given coordinates, and for randomly selecting failed package nodes based on given number of failed packages within a sub-region. In the discussion below, the term "M_fine" is used to describe the category of packages that fail initially, and "N_large" is used to describe the regions in which the remaining package failures occur. They are used as descriptive terms, and do not refer to the number of packages of either type that fail.

To perform the test, a routine called *test_ebs_random.f* Version 1.0, was written in Fortran 90. The core of the test routine is the EBS random release model, with code added to generate random node locations within a given field, loading nodes into an array containing node coordinates, and randomly assigning each node an index. When started, the user inputs the total number of nodes, number of sub-regions, random seed, number of M_fine individually failed packages, and number of species, and the on/off switch for releasing particles at the current time step. Then, the test routine generates randomly located nodes in each sub-region, loading those nodes into an array, and randomly assigns locations for the M_fine early failed packages. The test routine next reads in the controlling parameter for number of simulation time steps and the number of failed N_large packages in each sub-region at each time step. Once the nodes are generated and the parameters read in, *test_ebs_random.f* starts the core part of the calculation, the EBS random release model, to select nodes for the M_fine early failed packages and randomly select release nodes for the N_large failed packages in each sub-region.

To test the release model, a system was constructed consisting of two rectangular sub-regions defined by range of the (x,y) coordinates. Each sub-region has 20 nodes. Within each sub-region there are three early failed packages defined by location coordinates. A total of four time steps are simulated. At each time step, the number of N_large failed packages increase with time. The EBS random release model was used to select release nodes at each time step. The input parameters used for this test problem are listed in Table 8.

Table 8. List of Test Problem Input Parameters: EBS Random Release Model Verification

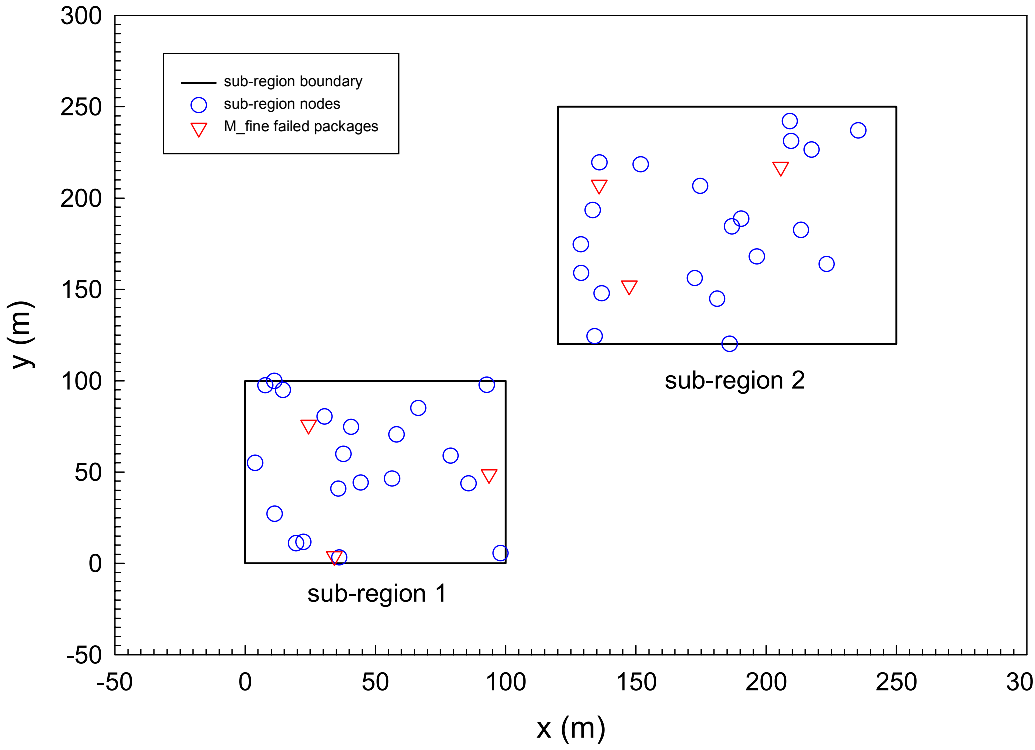
Parameters	Sub-region 1	Sub-region 2
Rectangular Sub-region Range	(x_min,x_max)=(0,100) (y_min,y_max)=(0,100)	(x_min,x_max)=(120,250) (y_min,y_max)=(120,250)
Number of Nodes	20	20
M_fine Early Failed Packages	3	3
Time Step 1: N_large Failed Packages	10	10
Time Step 2: N_large Failed Packages	15	15
Time Step 3: N_large Failed Packages	17	17
Time Step 4: N_large Failed Packages	20	20

The M_fine early failed package nodes and the randomly located N_large failed package nodes at different time steps are plotted in Figures 23 to 26.

At time step 1, there were 10 failed N_large packages in each sub-region containing 20 nodes. The EBS random release model should find nodes corresponding to the M_fine early failed packages and randomly select N_large number.

The results from this test indicate that showed that the algorithm successfully generates node locations and loads the generated nodes into the node coordinate array. The node index of the generated nodes are not in sequential order but rather randomly assigned to mimic the fact that node index for repository nodes may not be in sequential order. The test code also successfully generates 3 early failed package locations in each sub-region, as expected.

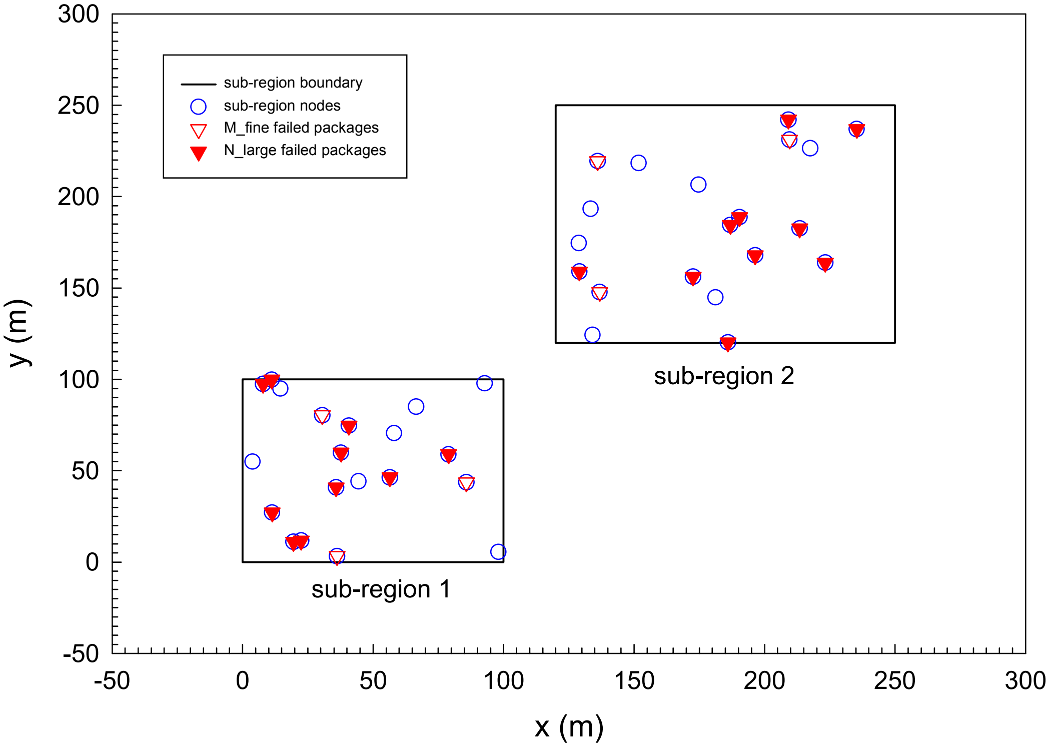
The selected M_fine early failed package nodes and the randomly located N_large failed package nodes at different time steps are plotted in Figures 23 to 26. Figure 23 shows the locations of nodes and the M_fine early failed packages.



DTN: LA0002BR12213S.002

Figure 23. Distribution of Nodes and M_fine Failed Packages in the Test Field

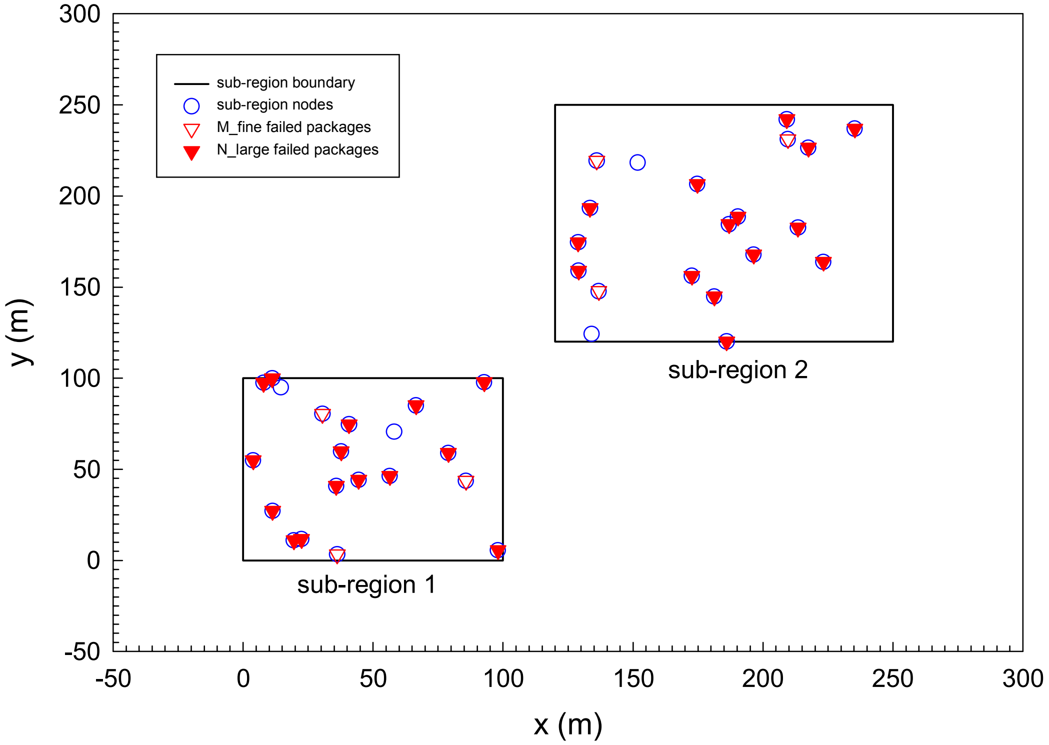
The EBS random release model was run to select nodes that are closest to the given locations of the early failed packages. At time step 1, there were 10 failed N_large packages in each sub-region containing 20 nodes. The EBS random release model should find nodes corresponding to the M_fine early failed packages and randomly select N_large number of nodes for the N_large failed packages in each sub-region. Figure 24 shows that the EBS random release model successfully selects nodes for the three early failed M_fine packages and randomly selects 10 nodes from the remaining 17 nodes for the N_large failed packages in each sub-region.



DTN: LA0002BR12213S.002

Figure 24. Distribution of Selected M_fine and N_large Failed Packages at Time Step 1

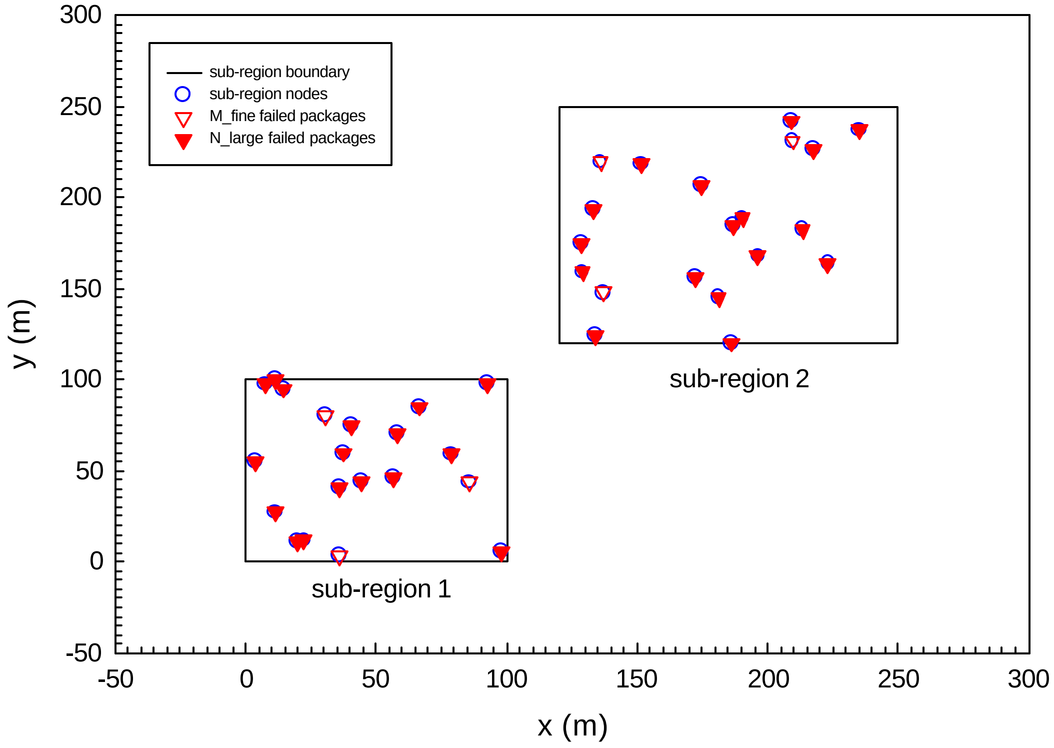
At time step 2, there are a total of 15 failed N_large packages in each sub-region. Thus, the number of newly failed N_large packages in each sub-region during this time step is five. The three early failed package nodes and the 10 previously selected nodes were not altered once they were selected. The EBS random release model then randomly selected five new nodes from the seven nodes remaining. Figure 25 shows the selected nodes failed during this time step.



DTN: LA0002BR12213S.002

Figure 25. Distribution of Selected M_fine and N_large Failed Packages at Time Step 2

At time step 3, the total number of failed N_large packages increased from 15 for the previous time step to 17 during the current time step. The number of newly failed N_large packages during this time step is two. The EBS random release model was used to select two nodes in each sub-region. Figure 26 shows the selected nodes during this time step.



DTN: LA0002BR12213S.002

Figure 26. Distribution of Selected M_fine and N_large Failed Packages at Time Step 3

At time step 4, although the total number of failed N_large failed packages was 20, there are no available nodes for selection at this time step. All nodes in each sub-region failed. The distribution of failed nodes at this time step is therefore same as that at time step 3. In summary, the results of this manual test indicates that the code is performing as expected and designed.

6.4 DUAL-PERMEABILITY BENCHMARKING SIMULATIONS

In Section 6.3 test cases were presented comparing the particle-tracking transport model against analytical solutions in a saturated, single-continuum system. Here a series of one-dimensional, dual-permeability calculations are carried out under unsaturated conditions. In all cases, the breakthrough curve at the water table is used to judge the adequacy of the model. Since the intended application for the UZ transport model is to simulate breakthrough curves at the water table, these tests exercise the code options in a manner equivalent to the way the code will be used in the TSPA calculations.

Analytical solutions for dual-permeability systems are difficult to obtain, even in one dimension, for any but the simplest cases. Therefore, the particle-tracking model for dual permeability, unsaturated flow and transport is compared to a discrete fracture model that uses FEHM's finite element transport module. In addition, simulation results are presented for the one-dimensional, dual-permeability model, also an option in FEHM, with transport solved using a finite element solution to the transport equation. These comparisons serve to illustrate the accuracy and, in some cases, assess the limitations of the particle-tracking model compared to other transport techniques.

In this suite of benchmarking runs, a one-dimensional system is selected because in the discrete fracture model, a one-dimensional, vertical fracture is easily discretized to capture the gradients in fluid pressure and solute concentration. The schematic in Figure 3 represents the model system geometry being used in these runs. The level of discretization used in this discrete fracture model is impossible to achieve in a site-scale model, but serves as an idealized benchmark for judging the suitability of a simplified particle-tracking model or other transport model. If the benchmarking is successful, the complexities in the three-dimensional flow system can be captured with a dual-permeability flow field, while the particle-tracking model can be used to capture the detailed physics of transport with matrix diffusion and sorption at the smaller scale. Our primary criterion for judging the adequacy of the particle-tracking model is that it adequately capture the behavior simulated in the discrete fracture model over a wide range of transport model parameters, for a wide range of flow system behavior. If the particle-tracking model can pass this test, its advantage of computational efficiency can be exploited in site-scale transport simulations. The purpose of also presenting one-dimensional, dual-permeability transport model results using a finite element solution is to demonstrate the relative ability of the more traditional model formulation to mimic the behavior of the discrete fracture model. Practically speaking, the dual-permeability transport simulator is one of the few alternative formulations available if a three-dimensional flow model is used directly for transport simulations.

The benchmarking system consists of a two-dimensional discrete fracture model grid that extends from an elevation of 630 m to 1150 m (meant to represent roughly the top of the TSw). The grid has a vertical spacing of 2.5 m, and a variable horizontal spacing that goes from less than 1 mm to about 0.9 m at the opposite side of the grid from the fracture. The model domain to the center line between fractures (*B* in Figure 3) is 5 m. This system is designed to simulate half of an open fracture and the adjacent matrix block out to the middle position between fractures (i.e. fracture spacing is 10 m).

Four flow situations are considered, each modeled by both the discrete-fracture and the dual-permeability models:

Case 1: Permeable fracture, impermeable matrix.

Case 2: Permeable fracture and matrix, with parallel flow in the fractures and matrix, but little or no fluid flow between the fracture and matrix from the radionuclide release point to the water table.

Case 3: Permeable fracture, varying fluid fluxes in the fractures and matrix in three hydrologic units. Transitions from fracture flow to matrix flow (and back) occur abruptly very close to the unit interfaces for the parameter values chosen.

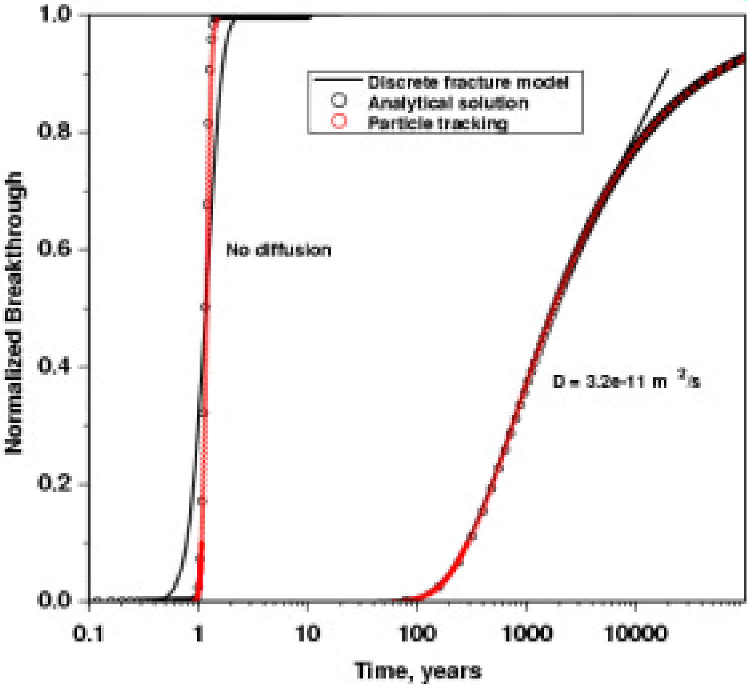
Case 4: Permeable fracture and matrix, with a flow restriction at the fracture/matrix interface such that fluid fluxes in the fractures gradually decrease with depth over the entire length of the model domain.

All cases use 5 mm/y as an average flux along the 5 m top surface. In the discrete fracture model, fluid is injected only in the fracture and allowed to imbibe into the matrix. At the radionuclide release level, water travels in parallel in the fracture and matrix (except for Case 4). The discrete fracture model domain was extended about 100 m below the assumed water table elevation of 730 m so as to not introduce any boundary-condition related artifacts at the 730 elevation. In the transport simulations, a row of nodes was selected 730 m, and a solute sink was placed there. Solute mass was introduced only at the fracture node at 1087.5 m, by fixing the concentration at a value of unity. The solute mass arrival rate at the water table plane is recorded in the solute mass balance output in the *.out file of FEHM. This time-varying curve was normalized by the injection solute mass rate. At long times, the input and output mass rates equal each other, and the breakthrough curve approaches unity.

For the one-dimensional, dual-permeability simulations, the same 5 mm/y rate was used, but in Cases 2 and 3 the flow distribution (fracture and matrix) was adjusted at the top of the model to ensure that the flow fraction at the release level mimicked the discrete fracture model. Particle-tracking model results in the form of arrival times for each particle at the water table are recorded in the *.fin file for post-processing to obtain the cumulative breakthrough curve. The one-dimensional, dual-permeability model simulations performed with the finite element transport simulation module are performed using the same flow fields as the particle-tracking models, and the post-processing is executed similarly to the discrete fracture model runs. The no-diffusion cases used 1.E-30 for the diffusion coefficient, while the diffusion cases assume 3.2E-11 m^2/s, and in some cases 1.0E-12 m^2/s. A longitudinal dispersivity of 1 m was assumed throughout the model domain in the vertical direction.

6.4.1 Case 1 Results

The first case isolates flow to only the fracture by applying an extremely low matrix permeability. In this way, the transport solution can be checked against the analytical solution for matrix diffusion. The analytical solution of Tang et al. (1981) is used for comparison of both the particle-tracking and discrete-fracture models. Figure 27 shows the breakthrough curves for the particle-tracking and discrete-fracture models compared to the analytical solution. With no diffusion into the matrix, breakthrough times are on the order of one year. The particle-tracking and analytical solutions agree very closely, showing that the particle-tracking model with dispersion is implemented properly. The deviation of the discrete fracture model from the analytical result is caused by numerical dispersion, but the breakthrough time agrees with the analytical solution.



DTN: LA0002BR12213S.002

Figure 27. Breakthrough Curves for Case 1 Comparing the Discrete Fracture Model with the Particle-tracking Method and an Analytical Solution

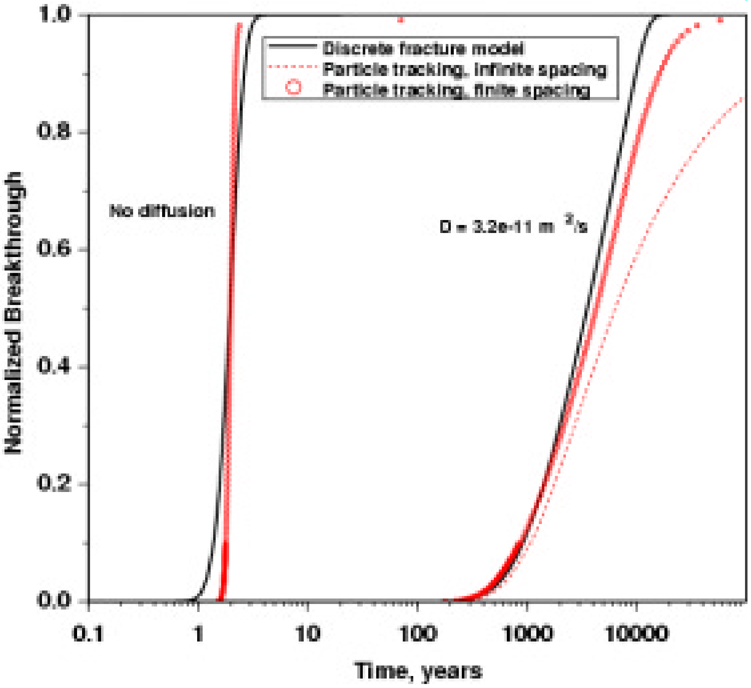
When diffusion into the stagnant fluid in the matrix is included, breakthrough times ranging from about 100 to 10,000 years are obtained, and again the particle-tracking and analytical solutions agree closely. The particle-tracking model follows the discrete fracture model closely for most of the breakthrough curve, deviating at long times. Since the infinite fracture spacing solution was used in this comparison, this departure at long times is expected because at late times, diffusion to the centerline between fractures will result in somewhat shorter travel times than if mass can diffuse greater distances away from the fracture than the half-spacing *B*. One factor to note about the particle-tracking solution is that the aperture required for the model to agree with the analytical solution under unsaturated conditions is the total aperture times the fracture saturation. This factor must be considered when using the particle-tracking model to perform unsaturated transport simulations.

This test case, simple enough to invoke an analytical solution, ensures that the numerical parameters associated with the discrete fracture model are suitable for performing comparisons to the particle-tracking model.

6.4.2 Case 2 Results

The second case considers parallel flow in the matrix and fracture. Water is injected at the top of the model only in the fracture node. It redistributes at steady state such that there is parallel flow in the fracture and matrix. This redistribution at the top occurs at a higher elevation than the repository horizon of 1087 m, so that the flow field relevant to transport is parallel flow in the fracture and matrix, with virtually no water flux between fracture and matrix along the transport pathway. Thus, the difference between Case 1 and Case 2 is that for Case 2, there is somewhat less flux in the fracture and some flux in the matrix, rather than stagnant fluid.

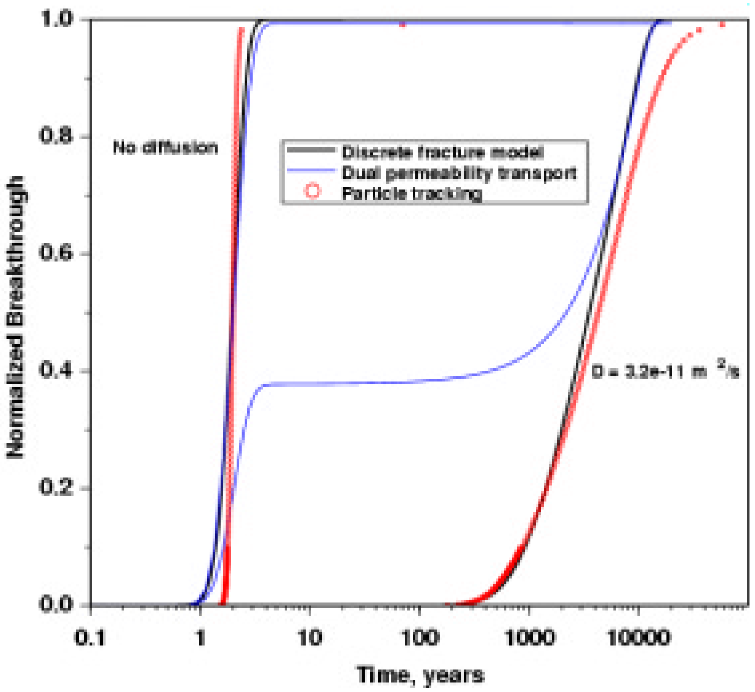
The discrete fracture and particle-tracking results without diffusion agree closely (Figure 28), as expected since this simulation isolates transport to the fractures, making it very similar to the no-diffusion results of Case 1. For a diffusion coefficient of 3.2E-11 m^2/s, three results are presented in Figure 28: the discrete fracture model, particle tracking with a infinite fracture spacing assumption, and particle tracking with the finite spacing assumption. The improved agreement of the finite spacing result (compared to the infinite spacing case) indicates that the matrix diffusion model with finite spacing captures the transport behavior more adequately than the infinite spacing case. The particle-tracking model agrees closely with the discrete fracture benchmark at early times, and departs from it somewhat at later times. This discrepancy is caused by the inherent assumption in the transfer function for matrix diffusion that the matrix fluid is stagnant. When enough solute diffuses into the matrix to result in travel times that approach the advective travel time in the matrix, the assumptions in the particle-tracking model are no longer strictly valid. Under those conditions, in the discrete fracture flow system, advection in the matrix carries solute to the outlet somewhat more rapidly than the particle-tracking model.



DTN: LA0002BR12213S.002

Figure 28. Breakthrough Curves for Case 2 Comparing the Discrete Fracture Model with the Particle-tracking Method

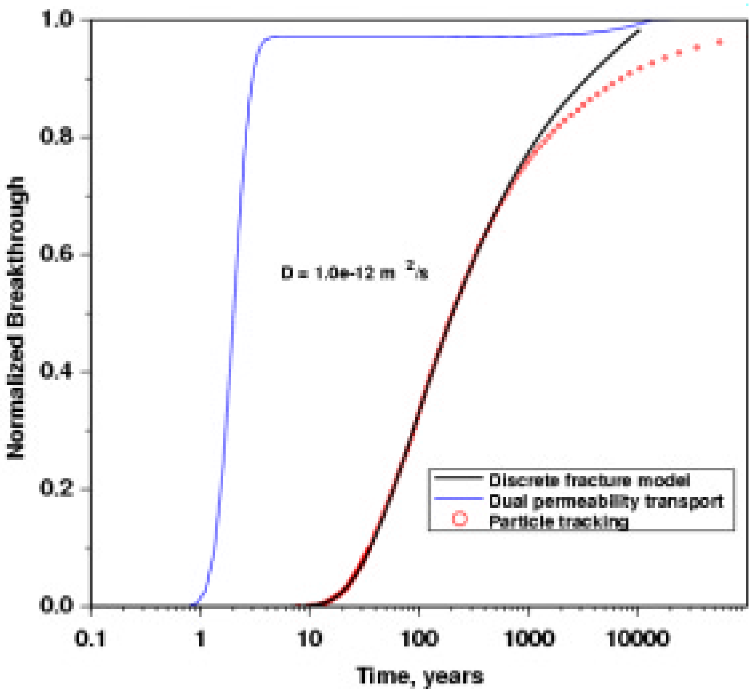
For comparison, the same simulation results are presented in Figure 29, but transport results using the finite element, dual-permeability transport module are also included. All of the models agree adequately without diffusion, but the one-dimensional, dual-permeability solution predicts much earlier breakthrough for this case. At early times, the dual-permeability solution fails to match the discrete fracture model because the fracture/matrix diffusion term is underestimated, and mass remains in the fracture to an unrealistic degree. Conversely, this model does a better job than the particle-tracking model at late times, fracture and matrix concentrations have equilibrated for this particular transport system. Nevertheless, the particle-tracking model adequately reproduces the discrete fracture model results over the entire range of travel times. When the diffusion coefficient is even smaller (Figure 30, diffusion coefficient of 1.0E-12 m^2/s), the finite element, dual-permeability transport model completely fails to capture the behavior of the discrete fracture model, whereas the particle-tracking model continues to provide a good representation of the behavior of the system.



DTN: LA0002BR12213S.002

Figure 29. Breakthrough Curves for Case 2 Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

To summarize this test case, systems with small to moderate amounts of diffusion are well captured (using the comparison to the discrete fracture model as a benchmark) by the particle-tracking algorithm, providing a superior transport model to a dual-permeability transport solution. The method avoids the artificial early-time breakthrough of mass that the dual-permeability transport solution yields due to the inability of the latter to capture sharp gradients in concentration away from the fractures at early times. At late times, the particle-tracking method deviates somewhat from the discrete fracture model. In estimates of unsaturated zone system performance, this deviation is probably more tolerable than the severe overestimate of early breakthrough of mass in the dual-permeability transport solution, but of course this factor must be judged in light of the goals of the model analysis being performed.



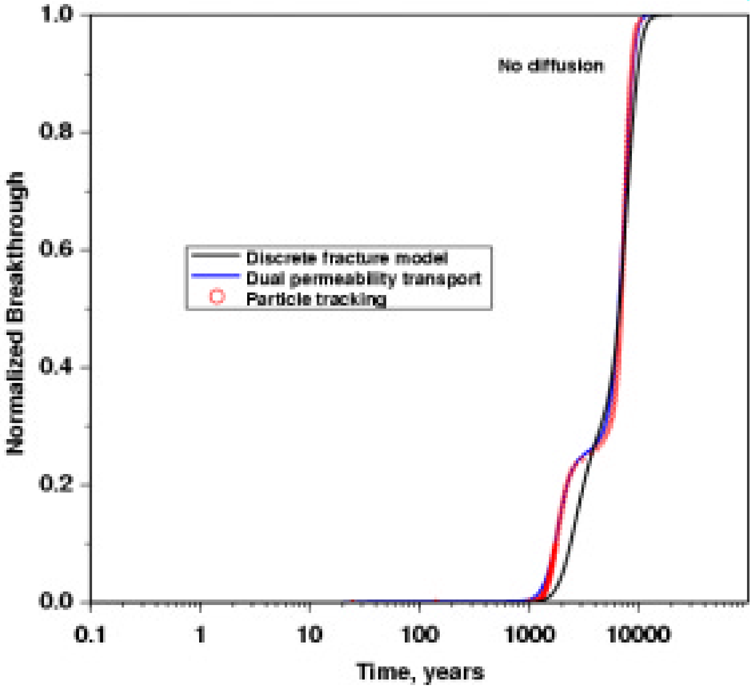
DTN: LA0002BR12213S.002

Figure 30. Breakthrough Curves for Case 2 (Low Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

6.4.3 Case 3 Results

This test case simulates the transition from predominantly fracture flow in the upper unit (given properties similar to the TSw) to predominantly matrix flow from an elevation of 898 m to 833.5 m (CHnv-like properties) to significantly more fracture flow from 833.5 m to the water table (CHnz-like properties). This case is realistic for testing the dual-permeability particle-tracking algorithm for a system containing several hydrogeologic units of contrasting properties, such as the Yucca Mountain unsaturated zone. To facilitate the model comparison, the discrete fracture model assumes identical properties within the entire length of the fracture, and does not attempt to account for contrasting fracture volume fraction and fracture spacing from unit to unit. Likewise, the dual-permeability model uses constant fracture properties throughout the model.

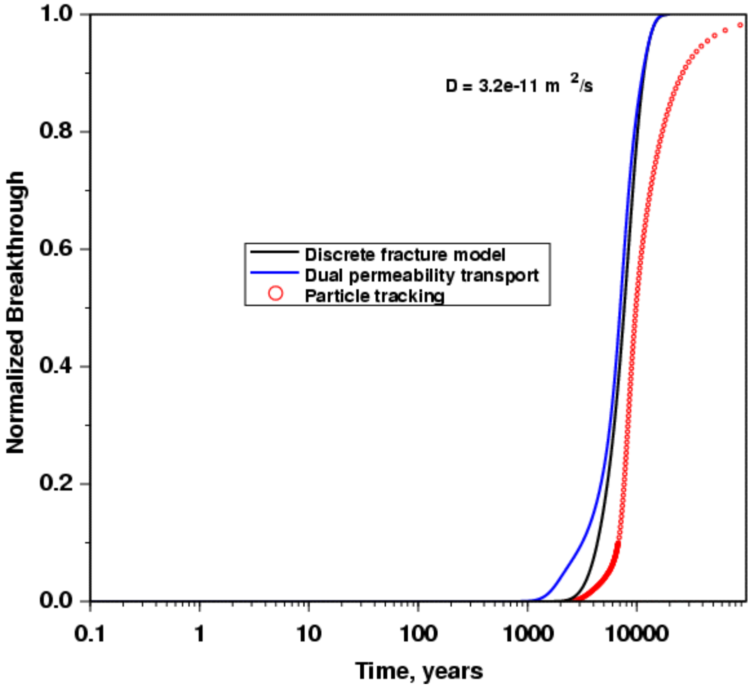
Figure 31 compares the simulation results for the three model options assuming no matrix diffusion. The particle-tracking and the dual-permeability finite element solution agree with one another closely, and compare fairly well with the discrete fracture model solution. The discrepancy with the discrete fracture simulation can be attributed to the inability to exactly match the advective velocities and detailed flow field of the discrete fracture model near the unit interfaces where transition from fracture to matrix (or matrix to fracture) flow occurs. This is a natural limitation of this benchmarking exercise for such a flow field, and cannot be easily overcome. Nevertheless, despite this disagreement, the characteristic behavior of the system is reproduced with both the dual-permeability transport model and the particle-tracking model. A minimum travel time through the system of the order of 1000 years is obtained due to matrix flow through the middle layer.



DTN: LA0002BR12213S.002

Figure 31. Breakthrough Curves for Case 3 (No Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

Figure 32 is the same comparison of the three models for a matrix diffusion coefficient of 3.2E-11 m^2/s. The particle-tracking model exhibits a somewhat later arrival time distribution than the discrete fracture model, whereas the finite element dual-permeability transport solution exhibits an earlier arrival. The dual-permeability transport solution probably better represents the overall breakthrough curve than does the particle-tracking result. However, the problem of rapid transport of some mass through the fractured units has not been eliminated (refer to Figure 29) but merely masked by the long travel time through the matrix-flow dominated unit. On the other hand, the particle-tracking solution tends to predict somewhat longer residence times than the discrete fracture model when diffusion is introduced. As in Case 2, this is the result of the assumption of diffusion into stagnant liquid as formulated using the transfer function approach. When diffusion is extensive enough to result in travel time delays that approach the travel time of fluid through the matrix block, the particle-tracking method as currently formulated will overestimate travel times. This conclusion is in contrast to the excellent agreement of the particle-tracking model to the discrete-fracture model for small to moderate amounts of diffusion.

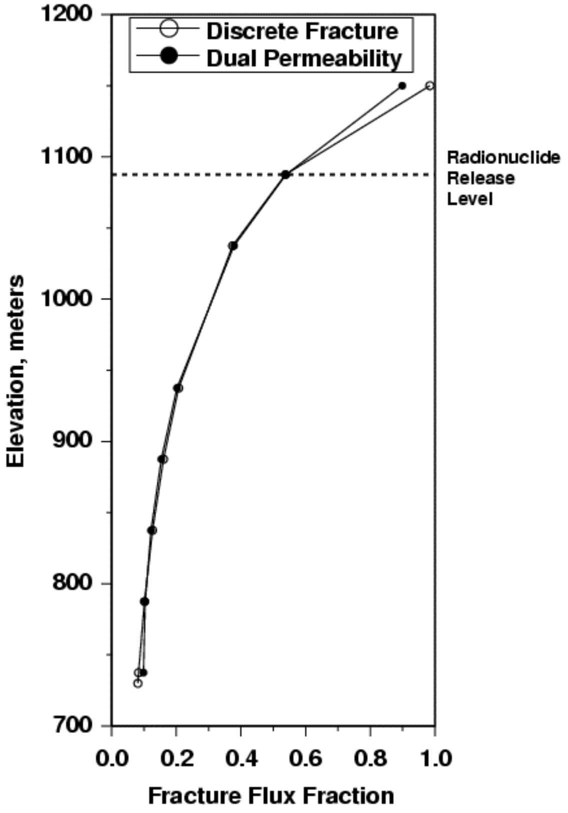


DTN: LA0002BR12213S.002

Figure 32. Breakthrough Curves for Case 3 (High Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

6.4.4 Case 4 Results

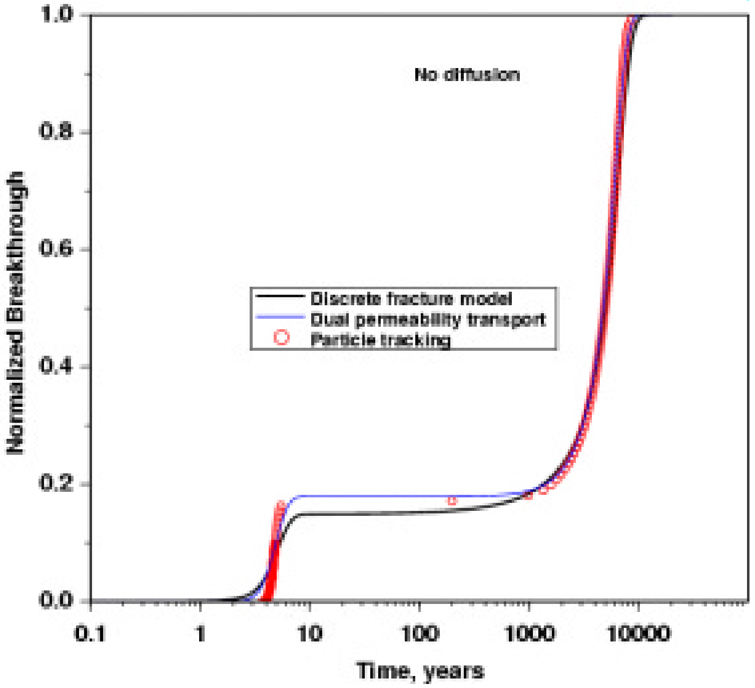
This test case is designed to examine a situation in which fluid is imbibed from the fracture to the matrix over the entire length of the fracture, including the portion of the model domain where the transport is occurring. The first step in developing this test case is to obtain discrete fracture and dual-permeability flow fields that match one another. Because our purpose is to compare transport models for comparable flow fields, the exact approach to obtain flow fields that agree is not important, as long as the same fluid flux values are achieved in the fracture and matrix at all locations in the model. The basic model consists of a single matrix unit with uniform properties, as in Case 2. To achieve the flow field for the discrete fracture model, a flow resistance was applied between the fracture nodes and the adjacent matrix, so that instead of redistributing within the top 10 to 20 meters of the model, flow imbibes from the fracture into the matrix along the entire length of the model. The dual-permeability flow field used parameters similar to those in Case 2 except for a larger matrix permeability and the injection of all inlet fluid into the fracture node instead of distributing it between the fracture and matrix. By simultaneously adjusting the matrix permeability in the dual-permeability model and the resistance factor in the discrete fracture model, comparable flow fields were obtained. Figure 33 shows the relative flow rate in the fracture portion of each model versus depth, illustrating that comparable flow fields were obtained for performing the transport runs.



DTN: LA0002BR12213S.002

Figure 33. Fracture Flux Fraction Versus Elevation for Case 4 Flow Fields: Discrete Fracture Model and One-Dimensional, Dual-Permeability Model

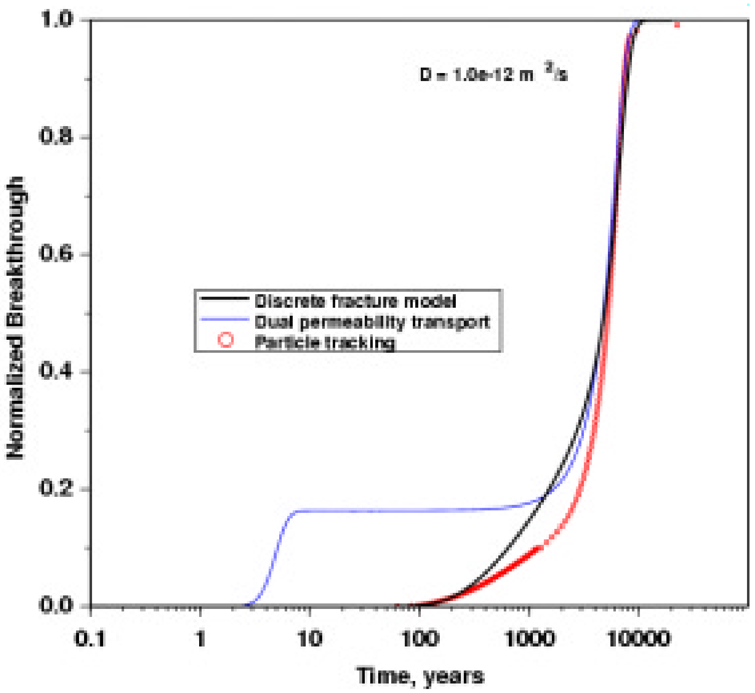
Figure 34 shows the results of the three transport models for the case of no diffusion. In all cases, early arrival times of a portion of the injected solute are observed due to the persistent fracture flow along the entire length of the system, and the absence of diffusion to bring mass into the matrix. The height of the plateau in the breakthrough curve represents the ratio of the flow rate of fluid reaching the water table in the fracture to the fracture flow rate at the radionuclide release location. The slight mismatch of the particle-tracking and dual-permeability transport models with the discrete fracture benchmark result is due to the difficulty of achieving an exact representation of the discrete fracture flow field with the one-dimensional, dual-permeability model. Nevertheless, the models are quite consistent with one another, illustrating that the particle-tracking model reproduces the idealized behavior of the system for this flow system for advective transport without diffusion.



DTN: LA0002BR12213S.002

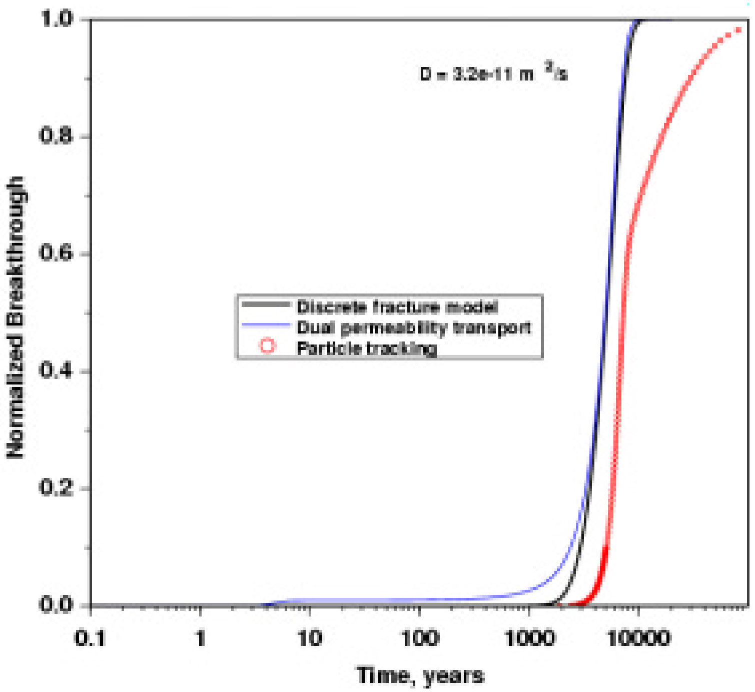
Figure 34. Breakthrough Curves for Case 4 (No Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

Figure 35 shows the results of the model for the low diffusion case (diffusion coefficient of 1.E-12 m^2/s), and Figure 36 compares the models for the higher diffusion coefficient of 3.2E-11 m^2/s. These figures suggest that the particle-tracking model continues to provide an adequate representation of the transport system for lower diffusion, but departs from the discrete-fracture model when higher diffusion is used. In contrast, the dual-permeability transport solution fails to capture the early-time behavior of the breakthrough curve, yielding an artificial early arrival of some solute mass. The extent of this problem is lessened for the higher diffusion coefficient case. Finally, Figure 37 shows the same model parameters as Figure 35, but includes sorption on the matrix rock (K_d = 5 mL/g). The early arriving mass continues to be present in the dual-permeability transport solution. Note the scale change in Figure 37 when evaluating the model behavior compared to that presented in Figure 35. The particle-tracking model also predicts an earlier first arrival of solute than the discrete fracture model result, but to a much lesser degree than the dual-permeability transport solution. The particle-tracking model could be improved in the future by adding a term to account for diffusive transport that carries mass into the matrix domain. For now, the transport result with diffusion and matrix sorption can be considered conservative for portions of a flow domain in which significant fracture-to-matrix flow occurs along the entire flow path. In dual-permeability models in which the transition occurs abruptly at the interface, the error will be isolated to nodes over which the flow transition occurs.



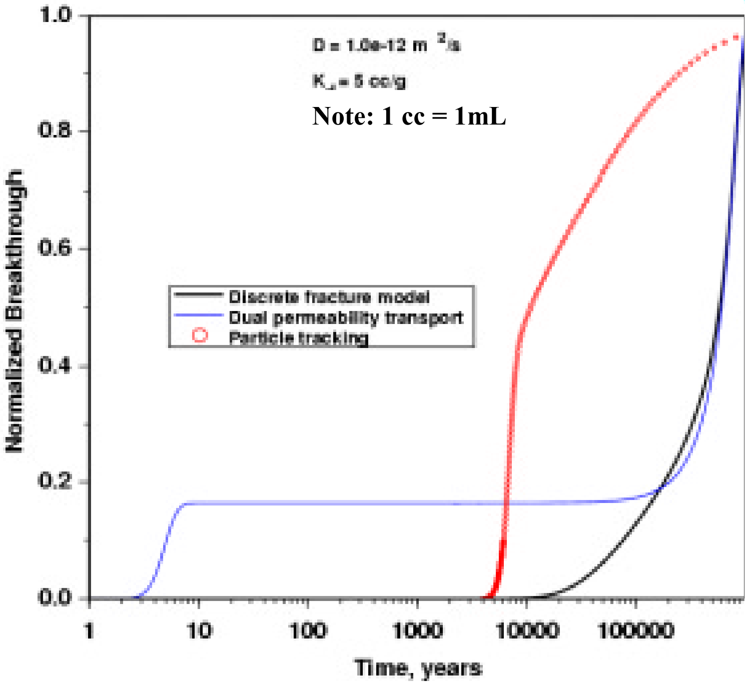
DTN: LA0002BR12213S.002

Figure 35. Breakthrough Curves for Case 4 (Low Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution



DTN: LA0002BR12213S.002

Figure 36. Breakthrough Curves for Case 4 (High Diffusion) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution



DTN: LA0002BR12213S.002

Figure 37. Breakthrough Curves for Case 4 (Low Diffusion, Matrix Sorption) Comparing the Discrete Fracture Model with the Particle-tracking Method and the Finite Element, Dual-Permeability Solution

7. CONCLUSIONS

The particle-tracking algorithm developed for this AMR incorporates the transport processes determined to be relevant in the site characterization program, including advection, dispersion, sorption, and matrix diffusion. In addition, new model development was required to allow for finite spacing between fractures in the matrix-diffusion model, multiple-species transport with decay/ingrowth, and the integration with the TOUGH2 and GoldSim applications. These capabilities were incorporated into the current version of FEHM, and validation testing demonstrated that the various processes are captured in the code. Therefore, this version of the code can be used to perform the UZ transport calculations for TSPA-SR. However, several caveats discussed below must be considered when executing the model runs to predict UZ radionuclide transport.

The particle-tracking method developed in this AMR, called the Residence Time Transfer Function (RTTF) particle-tracking technique, employs a new, efficient particle-tracking algorithm that is suitable for performing large-scale transport simulations. Like most particle-tracking methods, this algorithm provides advection-dominated solutions that are free of numerical dispersion. Thus, the technique can track a sharp front in solute concentration without the usual numerical dispersion or concentration-profile oscillations encountered in conventional finite-difference or finite-element solutions of the advection-dispersion equation.

The development that distinguishes the RTTF particle-tracking method from existing groundwater transport particle-tracking techniques is the conceptualization of transport as the movement of particles from cell to cell. This approach eliminates the need to resolve the velocity vectors by interpolation at all particle positions. Instead, the mean residence time and the probabilities of travel to adjacent cells are computed once for each cell (until the flow solution changes), regardless of how many particles travel through the cell. Another advantage of the cell-to-cell approach is that the information needed to compute particle residence times and pathways is readily available in finite-difference or finite-element solutions of the flow problem. The fluid mass and intercell mass flows rates are a direct result of the fluid flow solution, and thus the method can be implemented without regard to the nature of the numerical grid (structured versus unstructured grids, element shapes, etc.). The particle-tracking technique is also simple to implement for dual-permeability flow models, since the fracture-matrix interaction term is treated as another flow rate to or from a given node.

As particles are tracked through the model domain, no particle transport time stepping is necessary; all particles are tracked from the starting time to the ending time of a segment of the simulation. The only reasons to terminate and restart a simulation are either to update a flow field in a transient flow simulation or to pause to write the particle positions to output files. In the UZ transport application, the GoldSim application performs the time step control, and FEHM is called at each time step to compute the transport. The method is computationally efficient: simulations of several million particles are practical on conventional workstations without employing parallel processing hardware. Like most particle-tracking techniques, the algorithm would parallelize naturally, as the fate of each particle can be computed independently of the movement of other particles. The ability to track the movement of a large number of particles allows large-scale transport simulations to be carried out.

The concept of particle residence times in each cell has been extended to relax the assumption of pure plug flow without dispersion. Transfer functions have been developed to simulate dispersion, equilibrium sorption, matrix diffusion, and colloid-facilitated transport. For accuracy, dispersion coefficients must be small enough that the grid Peclet number remains greater than one throughout the grid. This limitation is not viewed as overly restrictive, because systems with large dispersion coefficients can be simulated accurately on relatively coarse grids with conventional finite-difference or finite-element solutions to the advection-dispersion equation. Thus, this particle-tracking method nicely complements the solute transport capabilities present in existing flow and transport codes such as FEHM. The matrix diffusion transfer function fills a void in our ability to simulate tracer experiments in fractured media using equivalent-continuum approaches. Transport time scales in tracer experiments are such that characteristic diffusional distances into the rock matrix are of the order of millimeters. The transfer function is a logical alternative to the explicit simulation of large concentration gradients in the matrix rock in the vicinity of fractures. Linear, equilibrium sorption is also incorporated into the matrix diffusion transfer function. This feature allows a variety of pollutant transport problems to be solved efficiently. When the assumptions inherent in the linear sorption isotherm are inappropriate, more complex chemical transport models should be employed.

The ability to implement the RTTF particle-tracking technique efficiently in a dual-permeability model framework is another advantage. Numerically, the communication between fractures and matrix is treated as one additional connection with a flow rate known from the fluid flow calculation. Rapid transit times through fractures can be duplicated easily using the method, whereas a typical equivalent-continuum representation will overestimate the travel time by using the matrix porosity in the computation. Finite-element solutions to the advection-dispersion equation in dual permeability, though possible, are problematic because the short travel times in fractures impose a very stringent constraint on the time step. Furthermore, our benchmarking exercise showed that significant error is introduced for diffusing species due to the lack of grid resolution in the matrix. In principle, a greater number of grid blocks could be placed in the matrix, but this solution is only practical for relatively small systems. It is not a reasonable alternative for site-scale models consisting of a large number of nodes. This limitation does not exist in the RTTF particle-tracking technique.

Like all numerical methods, the residence time/transfer function particle-tracking technique has limitations that must be considered when deciding whether its use is appropriate for a given application. The assumptions of advection-dominated transport and linear, equilibrium sorption have just been discussed. The possibility of grid orientation effects must also be considered. These effects can manifest themselves as an artificial lateral spreading of solute mass. However, for the UZ transport model application, the breakthrough of radionuclide mass at the water table is typically desired, rather than in situ concentration. A slight artificial spreading transverse to the flow direction would not be expected to change the travel time of an individual particle significantly.

As unstructured numerical grid generation techniques become more common, the simplicity of implementing the RTTF particle-tracking technique for these grids should be a great benefit. In the UZ, for example, unstructured grids are made to follow the complex stratigraphy present at the site, so that units with contrasting hydrologic properties can be represented. It may be that for these grids, orientation errors are small because the grid is aligned with the hydrostratigraphic units and thus are more likely to be aligned with the flow field. Thus, one of the RTTF particle-tracking technique's possible limitations should be minimized.

Finally, to judge the suitability of the particle-tracking model for unsaturated, dual-permeability transport, a benchmarking exercise was carried out comparing the particle-tracking and finite element, dual-permeability transport models to the presumably more representative discrete fracture model. The results of this suite of test cases highlighted the strengths and weaknesses of the particle-tracking model. The following conclusions were drawn from this analysis:

- Particle-tracking solutions that closely track the discrete fracture model result were obtained under all flow scenarios when matrix diffusion is not included.

- For low diffusion, the particle-tracking model reproduced the results of the discrete fracture model adequately for a variety of different flow scenarios. In contrast the dual-permeability transport solution yielded an artificial early arrival of solute for flow systems with significant fracture flow along the entire length of the fracture. For situations in which matrix-dominated flow occurred during a portion of the flow path, the early arrival is masked by the longer transport times within that unit, but the problem still exists.

- For higher diffusion, the particle-tracking model deviates from the discrete fracture model. This is not surprising, given the assumption used during the development of the method, whereby the fracture transport with matrix diffusion is conceptualized as essentially decoupled from the matrix flow and transport. Whether the model accurately represents reality depends on the actual behavior of the unsaturated zone, particularly with respect to the degree to which diffusion is capable to homogenize concentrations between the flowing fractures and the surrounding matrix.

- Not surprisingly, the dual-permeability transport model performs well in the high-diffusion regime. This result is due to the model formulation, consisting of a single matrix node for each fracture node. As in all such dual-porosity and dual-permeability models, accuracy in the low-diffusion regime is sacrificed in favor of a model that captures the long-time behavior of the system.

Given these results, it has been demonstrated that the particle-tracking model can be used in three-dimensional radionuclide transport simulations of the Yucca Mountain unsaturated zone as long as the limits on the model are recognized and parameters are chosen accordingly. Also, given the accuracy of the model without diffusion, supporting particle-tracking model runs in the absence of diffusion should be performed to access the importance of matrix diffusion to the overall conclusions of the performance assessment. In addition, an effort should be undertaken to relax the limiting assumptions that reduce the accuracy of the particle-tracking model when large amounts of diffusion are present. An appropriate gage of the extent of diffusion relative to the block size is a characteristic diffusion time to the center of a block. When the diffusion time is on the same order of magnitude or smaller than the transit time through the fracture, the fracture/matrix system in a dual-permeability model could be treated as an intimately coupled entity for the purposes of computing the particle transit time. This enhancement would do a better job at capturing the behavior in the presence of extensive diffusion without sacrificing the model’s ability to handle more moderate diffusion scenarios.

This document may be affected by technical product input information that requires confirmation. Any changes to the document that may occur as a result of completing the confirmation activities will be reflected in subsequent revisions. The status of input information quality may be confirmed by review of the Document Input Reference System database.

8. INPUTS AND REFERENCES

8.1 DOCUMENTS CITED

Akindunni, F.F.; Gillham, R.W.; Conant, B.; and Franz, T. 1995. "Modeling of Contaminant Movement Near Pumping Wells: Saturated-Unsaturated Flow with Particle Tracking." *Ground Water, 33,* (2), 264-274. Worthington, Ohio: National Water Well Association. TIC: 226455.

Bear, J.; Tsang, C.F.; and de Marsily, G., eds. 1993. *Flow and Contaminant Transport in Fractured Rock.* San Diego, California: Academic Press. TIC: 235461.

Chiang, C.Y.; Wheeler, M.F.; and Bedient, P.D. 1989. "A Modified Method of Characteristics Technique and Mixed Finite Elements Method for Simulation of Groundwater Solute Transport." *Water Resources Research, 25,* (7), 1541-1549. Washington, D.C.: American Geophysical Union. TIC: 226452.

Cordes, C. and Kinzelbach, W. 1992. "Continuous Groundwater Velocity Fields and Path Lines in Linear, Bilinear, and Trilinear Finite Elements." *Water Resources Research, 28,* (11), 2903-2911. Washington, D.C.: American Geophysical Union. TIC: 226449.

CRC 1991. *CRC Handbook of Chemistry and Physics, Lide, D.R., ed., 1991-1992.* 72nd Edition. Boca Raton, Florida: CRC Press. TIC: 3595.

CRWMS M&O 2000a. *Unsaturated Zone Flow and Transport Model and Abstractions for TSPA-SR Process Model Report.* TDR-NBS-HS-000002 REV 00. Las Vegas, Nevada: CRWMS M&O. ACC: MOL.20000320.0400.

CRWMS M&O 2000b. *Input Transmittal from LANL for Abstraction of Colloid-Facilitated Pu Transport Modeling for TSPA (ANL-NBS-HS-000031).* 00145T. Las Vegas, Nevada: CRWMS M&O. URN-0145.

CRWMS M&O 2000c. *Input Transmittal from LANL for Saturated Zone Transport Methodology and Transport Component Integration (MDL-NBS-HS-000010).* 00146T. Las Vegas, Nevada: CRWMS M&O. URN-0146.

CRWMS M&O 2000d. *Abstraction of Flow Fields for RIP (ID:U0125).* ANL-NBS-HS-000023 REV 00. Las Vegas, Nevada: CRWMS M&O. ACC: MOL.20000127.0089.

Desbarats, A.J. 1990. "Macrodispersion in Sand-Shale Sequences." *Water Resources Research, 26,* (1), 153-163. Washington, D.C.: American Geophysical Union. TIC: 236576.

DOE (U.S. Department of Energy) 1998. *Overview - Viability Assessment of a Repository at Yucca Mountain.* DOE/RW-0508. Washington, D.C.: U.S. Department of Energy, Office of Civilian Radioactive Waste Management. ACC: MOL.19981007.0027.

Fabriol, R.; Sauty, J.P.; and Ouzounian, G. 1993. "Coupling Geochemistry with a Particle Tracking Transport Model." *Journal of Contamination Hydrology, 13,* 117-129. New York, New York: Elsevier Science. TIC: 222395.

Freeze, R.A. and Cherry, J.A. 1979. *Groundwater.* Englewood Cliffs, New Jersey: Prentice-Hall. TIC: 217571.

Golder Associates 2000. *User's Guide GoldSim Graphical Simulation Environment*. Version 6.02. Draft #3. Redmond, Washington: Golder Associates Inc. TIC: 247324.

Iman, R.L. and Shortencarier, M.J. 1984. *A FORTRAN 77 Program and User's Guide for the Generation of Latin Hypercube and Random Samples for Use with Computer Models.* NUREG/CR-3624. Albuquerque, New Mexico: Sandia National Laboratories. TIC: 224592.

Johnson, J.A.; Ravi, V.; and Rumery, J.K. 1994. "Estimation of Solute Concentrations Using the Pathline Counting Method." *Ground Water, 32,* (5), 719-726. Worthington, Ohio: National Water Well Association. TIC: 226453.

Kinzelbach, W 1988. *The Random Walk Method in Pollutant Transport Simulation.* Groundwater Flow and Quality Modelling. p. 227-245. Norwell, Massachusetts: D. Reidel Publishing Company. TIC: 224064.

Latinopoulos, P. and Katsifarakis, K. 1991. "A Boundary Element and Particle Tracking Model for Advective Transport in Zoned Aquifers." *Journal of Hydrology, 124,* 159-176. Amsterdam, The Netherlands: Elsevier Science. TIC: 226466.

Liu, H.H.; Doughty, C.; and Bodvarsson, G.S. 1998. "An Active Fracture Model for Unsaturated Flow and Transport in Fractured Rocks." *Water Resources Research, 34,* (10), 2633-2646. Washington, D.C.: American Geophysical Union. TIC: 243012.

Lu, N. 1994. "A Semianalytical Method of Path Line Computation for Transient Finite-Difference Groundwater Flow Models." *Water Resources Research, 30,* (8), 2449-2459. Washington, D.C.: American Geophysical Union. TIC: 211997.

Maloszewski, P. and Zuber, A. 1991. "Influence of Matrix Diffusion and Exchange Reactions on Radiocarbon Ages in Fissured Carbonate Aquifers." *Water Resources Research, 27,* (8), 1937-1945. Washington, D.C.: American Geophysical Union. On Order Library Tracking Number-L1369

Marshall, T.J.; Holmes, J.W.; and Rose, C.W. 1996. Soil Physics, *Third Edition*. New York, New York: Cambridge University Press. TIC: 246638.

Nauman, E.B. and Buffham, B.A. 1983. *Mixing in Continuous Flow Systems.* New York, New York: John Wiley & Sons. TIC: 247226.

Neretnieks, I. 1980. "Diffusion in the Rock Matrix: An Important Factor in Radionuclide Retardation." *Journal of Geophysical Research, 85,* (B8), 4379-4397. Washington, D.C.: American Geophysical Union. TIC: 221345.

Neuman, S.P. 1984. "Adaptive Eulerian-Lagrangian Finite Element Method for Advection-Dispersion." *International Journal for Numerical Methods in Engineering, 20,* 321-337. New York, New York: John Wiley and Sons. TIC: 226440.

Pollack, D.W. 1988. "Semianalytical Computation of Path Lines for Finite-Difference Models." *Ground Water, 26,* (6), 743-750. Worthington, Ohio: National Water Well Association. TIC: 226464.

Pruess, K. 1991. *TOUGH2-A General-Purpose Numerical Simulator for Multiphase Fluid and Heat Flow.* LBL-29400. Berkeley, California: Lawrence Berkeley Laboratory. ACC: NNA.19940202.0088.

Reimus, P.W. 1995. *The Use of Synthetic Colloids in Tracer Transport Experiments in Saturated Rock Fractures.* Ph.D. Dissertation. LA-13004-T. Los Alamos, New Mexico: Los Alamos National Laboratory. TIC: 240694.

Reimus, P.W.; Adams, A.; Haga, M.J.; Humphrey, A.; Callahan, T.; Anghel, I.; and Counce, D. 1999. *Results and Interpretation of Hydraulic and Tracer Testing in the Prow Pass Tuff at the C-Holes.* Milestone SP32E7M4. Los Alamos, New Mexico: Los Alamos National Laboratory. TIC: 246377.

Robinson, B.A. 1994. "A Strategy for Validating a Conceptual Model for Radionuclide Migration in the Saturated Zone Beneath Yucca Mountain." *Radioactive Waste Management and Environmental Restoration, 19,* (1-3), 73-96. Yverdon, Switzerland: Harwood Academic Publishers. TIC: 222513.

Schafer-Perini, A.L. and Wilson, J.L. 1991. "Efficient and Accurate Front Tracking for Two-Dimensional Groundwater Flow Models." *Water Resources Research, 27,* (7), 1471-1485. Washington, D.C.: American Geophysical Union. TIC: 226448.

Scheibe, T.D. and Cole, C.R. 1994. "Non-Gaussian Particle Tracking: Application to Scaling of Transport Processes in Heterogeneous Porous Media." *Water Resources Research, 30,* (7), 2027-2039. Washington, D.C.: American Geophysical Union. TIC: 226446.

Starr, R.C.; Gillham, R.W.; and Sudicky, E.A. 1985. "Experimental Investigation of Solute Transport in Stratified Porous Media, 2. The Reactive Case." *Water Resources Research, 21,* (7), 1043-1050. Washington, D.C.: American Geophysical Union. TIC: 222358.

Sudicky, E.A. and Frind, E.O. 1982. "Contaminant Transport in Fractured Porous Media: Analytical Solutions for a System of Parallel Fractures." *Water Resources Research, 18,* (6), 1634-1642. Washington, D.C.: American Geophysical Union. TIC: 217475.

Tang, D.H.; Frind, E.O.; and Sudicky, E.A. 1981. "Contaminant Transport in Fractured Porous Media: Analytical Solution for a Single Fracture." *Water Resources Research, 17,* (3), 555-564. Washington, D.C.: American Geophysical Union. TIC: 225358.

Tompson, A.F.B. and Gelhar, L.W. 1990. "Numerical Simulation of Solute Transport in Three-Dimensional, Randomly Heterogeneous Porous Media." *Water Resources Research, 26,* (10), 2541-2562. Washington, D.C.: American Geophysical Union. TIC: 224902.

van Genuchten, M.T. 1980. "A Closed-Form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils." *Soil Science Society American Journal, 44,* (5), 892-898. Madison, Wisconsin: Soil Science Society of America. TIC: 217327.

van Genuchten, M.T. 1985. "Convective-Dispersive Transport of Solutes Involved in Sequential First-Order Decay Reactions." *Computers in Geosciences, 11,* (2), 129-147. New York, New York: Pergamon Press. TIC: 245188.

Wolfsberg, A.V. and Freyberg, D.L. 1994. "Efficient Simulation of Single Species and Multispecies Transport in Groundwater with Local Adaptive Grid Refinement." *Water Resources Research, 30,* (11), 2979-2991. Washington, D.C.: American Geophysical Union. TIC: 226450.

Yamashita, R. and Kimura, H. 1990. "Particle-Tracking Technique for Nuclide Decay Chain Transport in Fractured Porous Media." *Journal of Nuclear Science and Technology, 27,* (11), 1041-1049. Tokyo, Japan: Nihon Genshryoku Gakkai. TIC: 236573.

Yeh, G.T. 1990. "A Lagrangian-Eulerian Method with Zoomable Hidden Fine-Mesh Approach to Solving Advective-Dispersion Equations." *Water Resources Research, 26,* (6), 1133-1144. Washington, D.C.: American Geophysical Union. TIC: 224067.

Zheng, C. 1993. "Extension of the Method of Characteristics for Simulation of Solute Transport in Three Dimensions." *Ground Water, 31,* (3), 456-465. Worthington, Ohio: Water Well Publishing Company. TIC: 226465.

Zheng, C. 1994. "Analysis of Particle Tracking Errors Associated with Spatial Discretization." *Ground Water, 32,* (5), 821-828. Dublin, Ohio: Water Well Publishing Company. TIC: 226463.

Zimmerman, R.W.; Chen, G.; Hadgu, T.; and Bodvarsson, G.S. 1993. "A Numerical Dual-Porosity Model with Semianalytical Treatment of Fracture/Matrix Flow." *Water Resources Research, 29,* (7), 2127-2137. Washington, D.C.: American Geophysical Union. TIC: 236571.

Zyvoloski, G.; Dash, Z.; and Kelkar, S. 1992. *FEHMN 1.0: Finite Element Heat and Mass Transfer Code.* LA-12062-MS, Rev. 1. Los Alamos, New Mexico: Los Alamos National Laboratory. ACC: NNA.19910625.0038.

Zyvoloski, G.A.; Robinson, B.A.; Dash, Z.V.; and Trease, L.L. 1997. *Summary of Models and Methods for the FEHM Application - A Finite Element Heat- and Mass-Transfer Code.* LA-13307-MS. Los Alamos, New Mexico: Los Alamos National Laboratory. TIC: 235587.

8.2 CODES, STANDARDS, REGULATIONS, AND PROCEDURES

AP-3.10Q, Rev. 2, ICN 0. *Analysis and Models*. Washington, D.C.: DOE OCRWM. ACC: MOL.20000217.0246.

AP-SI.1Q, Rev 2, ICN 4. OCRWM Procedure: Software Management. Washington, D.C.: DOE OCRWM. ACC: MOL.20000223.0508.

CRWMS M&O 1999a. *Improve Documentation and Verification of Radionuclide Decay Model (Rev. 01). ID: U3050, Activity: SPP5190.* Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19990707.0132.

CRWMS M&O 1999b. *Enhance Particle Tracking Features for TSPA (Rev. 01). ID: U3050, Activity: SPP5190.* Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19990707.0134.

CRWMS M&O 1999c. *Improve Matrix Diffusion Model (Rev. 01). ID: U3050, Activity: SPP5190.* Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19990707.0137.

CRWMS M&O 1999d. *Conduct of Performance Assessment.* Activity Evaluation, September 30, 1999. Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19991028.0092.

DOE (U.S. Department of Energy) 2000. *Quality Assurance Requirements and Description.* DOE/RW-0333P, Rev. 9. Washington, D.C.: U.S. Department of Energy, Office of Civilian Radioactive Waste Management. ACC: MOL.19991028.0012.

Dyer, J.R. 1999. "Revised Interim Guidance Pending Issuance of New U.S. Nuclear Regulatory Commission (NRC) Regulations (Revision 01, July 22, 1999), for Yucca Mountain, Nevada." Letter from Dr. J.R. Dyer (DOE/YMSCO) to Dr. D.R. Wilkins (CRWMS M&O), September 3, 1999, OL&RC:SB-1714, with enclosure, "Interim Guidance Pending Issuance of New NRC Regulations for Yucca Mountain (Revision 01)." ACC: MOL.19990910.0079.

QAP-2-0, Rev. 5. *Conduct of Activities*. Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19980826.0209.

QAP-2-3, Rev. 10. *Classification of Permanent Items*. Las Vegas, Nevada: CRWMS M&O. ACC: MOL.19990316.0006.

8.3 SOFTWARE

Los Alamos National Laboratory. *Software Code: FEHM V2.10.* 1999. V2.10. SUN Ultra Sparc, PC. 10086-2.10-00.

Lawrence Berkeley National Laboratory. *Software Code: FEHM V2.00.* 2000. V2.00. SUN Ultra Sparc. 10031-2.00-00.

8.4 SOURCE DATA, LISTED BY DATA TRACKING NUMBER

GS950608312231.008. Moisture Retention Data from Boreholes USW UZ-N27 and UE-25 UZ#16. Submittal date: 06/06/1995.

GS960808312231.003. Moisture Retention Data for Samples from Boreholes USW SD-7, USW SD-9, USW SD-12 and UE-25 UZ#16. Submittal date: 08/30/1996.

GS980908312242.039. Unsaturated Water Retention Data for Lexan-Sealed Samples from USW SD-6 Measured Using a Centrifuge. Submittal date: 09/22/1998.

LA0002PR831231.003. Probabilities from C-Wells Microsphere Data. Submittal date: 02/17/2000.

LA0003MCG12213.002. Cumulative Probabilities for Colloid Transport Between One Matrix and Another Calculated from Interpolation of Pore Volume Data from Yucca Mountain Hydrologic (Stratigraphic) Samples. Submittal date: 03/10/2000.

LB990501233129.001. Fracture Properties for the UZ Model Grids and Uncalibrated Fracture and Matrix Properties for the UZ Model Layers for AMR U0090, "Analysis of Hydrologic Properties Data". Submittal date: 08/25/1999.

ATTACHMENT I. SOFTWARE ROUTINE TEST

Software routine TEST, Version 1.0 was used to compare number of decayed particles calculated based on the direct discrete approach and the integration approach. The algorithm used in the code is listed below:

```
Input output file names for storing calculation data
Input simulation start time and end time
Input simulation time interval
Input particle injection start time and end time
Input particle half life
Input number of particles per node
Input total number of nodes
Calculate number of simulation time steps (nt) and decay rate
Calculate integration time interval
Loop 1, for time step=1 to nt
   Calculate number of particles decayed per node using the integration approach in Eq. (2)
   Calculate total number of particles decayed over all nodes
   Loop 2, for all particles over a node
        Calculate number of particles decayed on the node using the discrete approach
            based on Eq. (1)
   End loop 2
   Calculate total number of particles decayed over all nodes
   Calculate integration % error compared to discrete formula
   Record the maximum % error and the corresponding time
   Output the results at each time step
   Increase the time by the simulation time interval
End loop 1
Output the maximum % error, the corresponding time, and the values calculated based on the
    discrete formula and the integration approach.
End software routine TEST
```

The software routine TEST is attached for reference. The software routine was compiled and run on a SunUltra Sparc work station with SunOS 5.5 C++ compiler (property tag: R431923, Sandia National Laboratories, Albuquerque, New Mexico).

To comply with the requirements of AP-SI.1Q on software routines, routine TEST was checked with hand calculations.

The test case had 1 node with 10 particles uniformly released during the time period of 0 to 9 years, the half life of the species was 10 years. Calculation results from TEST are attached. The hand calculation results at t=30 years were used to check the corresponding software routine results.

Table I-1 lists the results of hand calculations and the corresponding results of software routine TEST. A comparison between those results showed good agreement and proved that the software routine TEST was correct.

Table I-1. Results From Hand Calculations and Software Routine TEST

Particle Index	Release Time, t_i (years)	**Time, t=30**			
		Discrete Approach		Integration Approach	
		Calculation $1-\exp(-\lambda(t-t_i))$	TEST	Calculation $\{(\tau_1-\tau_2)+[\exp(-\lambda\tau_1)-\exp(-\lambda\tau_2)]/\lambda\}/\Delta t$	TEST
1	0	0.875000		$\tau_1=t-t_1=30-0=30$ $\tau_2=t-t_{10}=30-9=21$ $\lambda=0.06931472$ $\Delta t=9/10=0.9$ $(\tau_2-\tau_1)=30-21=9$ $\exp(-\lambda\tau_1)=0.12500$ $\exp(-\lambda\tau_2)=0.23326$	
2	1	0.866028			
3	2	0.856412			
4	3	0.846106			
5	4	0.835061			
6	5	0.823223			
7	6	0.810535			
8	7	0.796936			
9	8	0.782362			
10	9	0.766741			
Total: sum=sum+0.5		8.758409	8.81819	8.76456	8.76463
int(sum)		8	8	8	8

List of Software Routine TEST Output Results for the Test Problem

```
input simulation start time and end time
0     40
input simulation time interval dt
10
The particle injection start time and end time
0     9
The half life
10
number of particles released per node
10
Total number of nodes
1
At time=0
Integration Decayed: 0
Discrete    Decayed: 0
Integration Percent error: 0

At time=10
Integration Decayed: 3
Discrete    Decayed: 3
Integration Percent error: 0

At time=20
Integration Decayed: 7
Discrete    Decayed: 7
Integration Percent error: 0

At time=30
Integration Decayed: 8
Discrete    Decayed: 8
Integration Percent error: 0



Over Results:
Discrete == Integration
```

List of Software Routine TEST

```
//start of code TEST

#include <stdio.h>
#include <iostream.h>
#include <fstream.h>
#include <math.h>
#include <stdlib.h>

main()
{
int i,j,k,l,ii,nt;
char outfile[60],errfile[60];
double t1,t2,tt,tot,half,t,m,dt,time,ti2,ti1,np;
double tot1,pert=0.,dtime,endtime,maxerror=0.0;
double tmptime=0.,tmptot=0.,tmptot2=0.;

cout<<"Output file name for storing calculation results"<<endl;
cin>>outfile;


cout<<"Output file name for storing %error data"<<endl;
cin>>errfile;

ofstream outFile (outfile, ios::out);
if(!outFile)
{
  cerr<<"Can not open file:"<<outfile<<endl;
  exit(-1);
}


ofstream outErr (errfile, ios::out);
if(!outErr)
{
  cerr<<"Can not open file:"<<errfile<<endl;
  exit(-1);
}

cout<<"input simulation start time and end time"<<endl;
cin>>time>>endtime;
cout<<"input time interval dt"<<endl;
cin>>dtime;
cout<<"The injection start time and end time "<<endl;
cin>>t1>>t2;
cout<<"The half life"<<endl;
cin>>half;
cout<<"number of particles released per node"<<endl;
cin>>np;
cout<<"Total number of nodes"<<endl;
cin>>m;

half=0.6931472/half; //degration rate
```

```
nt=(endtime-time)/dtime;
dt=(t2-t1)/np;


//calculate # of particles decayed using the integration method

for(ii=1;ii<=nt;ii++)
{
  cout<<"At time="<<time<<endl;
  outFile<<"At time="<<time<<endl;
  ti2=time-t2;
  if(ti2<0.)ti2=0.;
  ti1=time-t1;
  if(ti1<0.)ti1=0.;
  cout<<"time1="<<ti1<<endl;
  cout<<"time2="<<ti2<<endl;
  tot=ti2-ti1+(exp(-half*ti2)-exp(-half*ti1))/half;
  tot=-tot*np*m/(t2-t1);
  tot1=tot;
  tot1=int(tot1+0.5);
  cout<<"ivdt="<<np*m/(t2-t1)<<endl;
  cout<<"kfact="<<half<<endl;
  cout<<"ti2="<<ti2<<endl;
  cout<<"Integration: tot="<<tot1<<endl;
  outFile<<"Integration Decayed: "<<tot1<<endl;

//end of integration calculation.

//calculate # of particles decayed using discrete method.

  tot=0;
  for(j=1;j<=np;j++)
  {
    t=t1+(j-1)*dt;
    tt=time-t;
    if(tt>=0.)
    tot +=(1-exp(-half*(time-t)));
  }
  tot=tot*m;
  tot=int(tot+0.5);
  cout<<"Discrete: tot="<<tot<<endl;
  outFile<<"Discrete    Decayed: "<<tot<<endl;
  cout<<"final time="<<t<<endl;
  if(tot != 0.)pert=(100.*(tot1-tot)/tot);
  cout<<"Percent error: "<<pert<<endl;
  outFile<<"Integration Percent error: "<<pert<<endl;
  outFile<<endl;
  outErr<<tot<<"\t"<<pert<<endl;
  if(pert <0.)pert=-pert;

//find the maxium percentage error.

  if(pert > maxerror)
    {
      maxerror=pert;
      tmptime=time;
      tmptot=tot1;
      tmptot2=tot;
      }
  if(tot1 >=np*m)
    {
      cerr<<"Integration larger than the original"<<endl;
      cerr<<"tot1="<<tot1<<" tot="<<tot<<endl;
      cout<<"The max error is: "<<maxerror<<endl;
      cout<<"happened at time: "<<tmptime<<endl;
      cout<<"Integral:  "<<tmptot<<"  Discrete:  "<<tmptot2<<endl;
      return tot1;
    }
  time=time+dtime;
}

//end of discrete calculation.
if(maxerror==0.0)
  {
    cout<<"\n\nOver Results:"<<endl;
    cout<<"Discrete == Integration"<<endl;
    outFile<<"Discrete == Integration"<<endl;
  }
  else
  {
    cout<<"\n\nOverall Results:\n"<<endl;
    cout<<"The max error is % : "<<maxerror<<endl;
    cout<<"happened at time   : "<<tmptime<<endl;
    cout<<"Integral:  "<<tmptot<<"  Discrete:  "<<tmptot2<<endl;
    outFile<<"\n\nOverall Results:\n"<<endl;
    outFile<<"The max error is % : "<<maxerror<<endl;
    outFile<<"happened at time   : "<<tmptime<<endl;
    outFile<<"Integral:  "<<tmptot<<"  Discrete:  "<<tmptot2<<endl;
    outFile<<endl;
```

ATTACHMENT II. FEHM POST PROCESS SOFTWARE ROUTINE BKPM

Software routine BKPM, Version 1.0 was written in FORTRAN 77 and was use to extract particle mass flow rates, fractions of particles flow out of the system during the current time step, and normalized cumulative breakthrough data at the out flow boundary. Software routine BKPM was developed and installed on a Sun UltrSparc with SunOS 5.6 (CPUid: R431923) at Sandia National Laboratories in Albuquerque, New Mexico.

Figure II-1 is the flow chart of BKPM used for extracting transport information from FEHM particle output files.

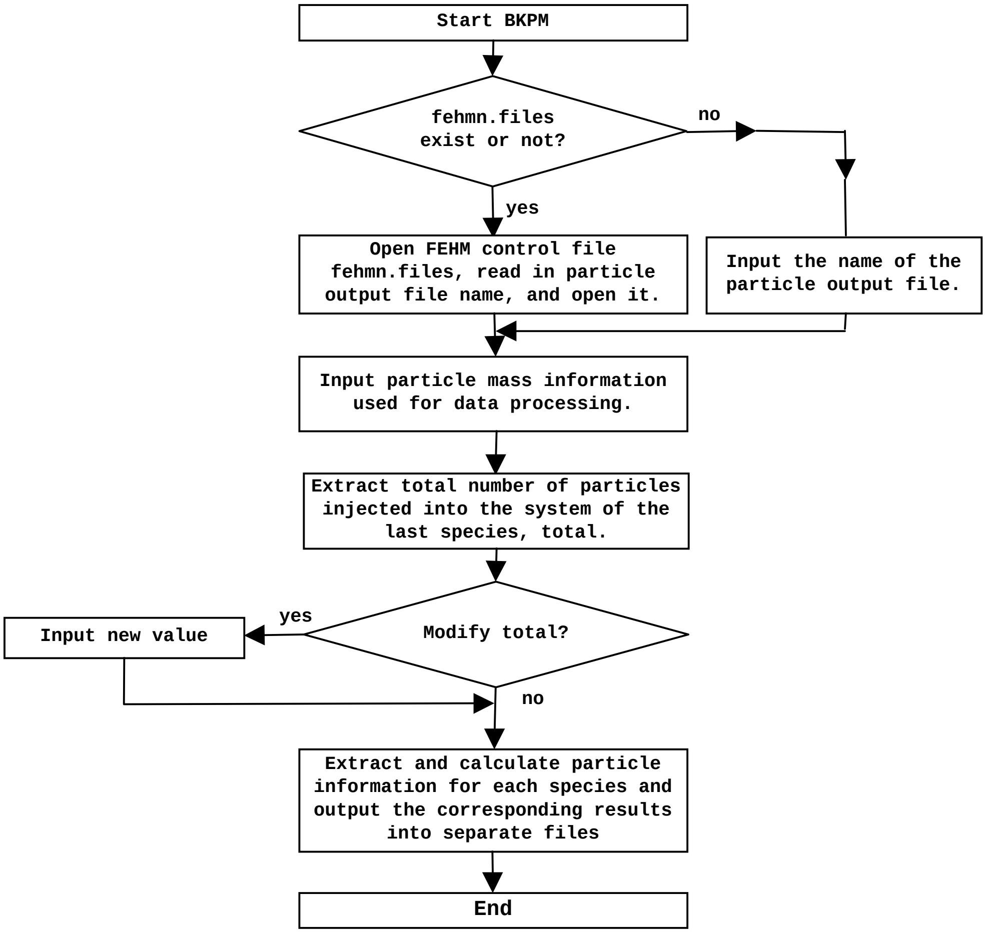

Start BKPM
fehmn.files exist or not?
no
yes
Open FEHM control file fehmn.files, read in particle output file name, and open it.
Input the name of the particle output file.
Input particle mass information used for data processing.
Extract total number of particles injected into the system of the last species, total.
Modify total?
yes
Input new value
no
Extract and calculate particle information for each species and output the corresponding results into separate files
End

Figure II-1. Flow Chart of Software Routine BKPM

When started, software routine BKPM will first search under current directory for FEHM control file *fehmn.files*. If the file exists, the routine opens the control file and reads in the particle output file name which is listed in the control file. If the routine can not find FEHM control file *fehmn.files*, then the routine asks the user to input the particle output file name.

In order for the post processor to calculate the mass flow rate, the user needs to input the amount of mass released at the source. The routine opens the particle output file, reads in the particle information, and extracts and prints out the total number of particles injected into the system, *total*, for the last species. Value *total* is used in the routine to calculate the normalized cumulative breakthrough data for each species. The user can alter the value of *total* by inputting a new value.

For each time step, the routine calculates particle mass flow rates, fractions of particles flow out of the system during the current time step, and normalized cumulative breakthrough curves at the out flow boundary for each species based on Equations (II-1), (II-2), and (II-3), respectively.

$$mass\_flux\_i = \frac{n_{itc} - n_{itp}}{t_c - t_p} \qquad \text{(Eq. II-1)}$$

$$fractions\_left\_system\_i = \frac{n_{itc} - n_{itp}}{total} \qquad \text{(Eq. II-2)}$$

$$Normalized\_cumulative\_i = \frac{n_{itc}}{total} \qquad \text{(Eq. II-3)}$$

where *i* is the species index. Parameters n_{tc} and n_{tp} are the total number of particles have left the system from the start of the simulation to current time t_c and to previous time t_p respectively.

The representative time for breakthrough data during the current time step is

$$reptime = t_p + \frac{t_c - t_p}{2} \qquad \text{(Eq. II-4)}$$

The corresponding results for each species are written to default files*: spec_1, spec_2, ...,* and *spec_n*, respectively, for a total of n species. Each output file contains the following information: time (year), mass flux (particles/year)/relative concentration, normalized cumulative breakthrough, and fraction of particles left during current time step.

Software routine BKPM was tested using part of a FEHM particle output file *out1.out_test*.

The test case is for a 4 member sequential decay chain (species 1 decays into 2, species 2 decays into 3, and species 3 decays into 4, while species 4 decays but without ingrowth) plus one conservative species advection-dispersion transport in a 1-D system. The initial concentrations of all species were 0. A total of 1 mole of species 1, represented by 500,000 particles, were injected into the system uniformly from time 0 to 5,000 years. There was no source release for species 2, 3, and 4. The conservative species, 5, was used in the test case as a reference. As was done for species 1, 500,000 particles representing 1 mole of species 5 were injected into the system during the same injection period.

Code FEHM v2.0 was run to simulate the transport process with decay-ingrowth. FEHM generated results were then used to test the post process routine BKPM.

Files used in the test are listed below. Those files include FEHM control file: *fehmn.files* and particle output file: *out1.out_test*. The source code of BKPM is listed at the end of Attachment II for reference.

FEHM control file *fehmn.files* contains input and output file names used in FEHM particle simulations. Among them, the fourth line lists the particle output file name (in the example *fehmn.files* below, it is *out1.out_test*). The example contents of *fehmn.files* are shown below.

Contents of example FEHM control file *fehmn.files*.

```
inp1.in
grid
inp1.in
out1.out_test
inp1.ini
inp1.fin
out1.his
out1.trc
inp1.con



all
0
```

When started, routine BKPM automatically opens file *fehmn.files*, reads in particle output file name *out1.out_test*, and opens it for data processing. Particle output file *out1.out_test* contains FEHM run time information on input and output files used, FEHM modules activated, memory required for storing major parameters, mass and energy balance information when flow calculations are carried out, and particle mass balance information at different time steps. As the goal of routine BKPM is to extract particle mass flow information and calculate particle breakthrough curves at the outflow boundary, routine BKPM only reads in and processes particle information and skips through the rest of the information in the file. The contents of file *out1.out_test* are shown below.

Contents of file *out1.out_test*

```
FEHM V2.00 99-05-03    06/14/1999    09:30:05

 File purpose - Variable - Unit number - File name

    control       - iocntl -  1 - fehmn.files
    input         - inpt   - 11 - inp1.in
    geometry      - incoor - 12 - grid
    zone          - inzone - 11 - inp1.in
    output        - iout   - 14 - out1.out
    initial state - iread  - 15 - inp1.ini
    final state   - isave  - 16 - inp1.fin
    time history  - ishis  - 17 - out1.his
    time his.(tr) - istrc  - 18 - out1.trc
    contour plot  - iscon  - 19 - inp1.con
    con plot (dp) - iscon1 -  0 - not using
    fe coef stor  - isstor -  0 - not using
    input check   - ischk  - 22 - fehmn.chk
 Value provided to subroutine user: not using

 mptr read from optional input file: inp.ptrk

 Check of FEHMN against SORBEQ, All isotherms



 File purpose - Variable - Unit number - File name

    control       - iocntl -  1 - fehmn.files
    input         - inpt   - 11 - inp1.in
    geometry      - incoor - 12 - grid
    zone          - inzone - 11 - inp1.in
    output        - iout   - 14 - out1.out
    initial state - iread  - 15 - inp1.ini
    final state   - isave  - 16 - inp1.fin
    time history  - ishis  - 17 - out1.his
    time his.(tr) - istrc  - 18 - out1.trc
    contour plot  - iscon  - 19 - inp1.con
    con plot (dp) - iscon1 -  0 - not using
    fe coef stor  - isstor -  0 - not using
    input check   - ischk  - 22 - fehmn.chk
 Value provided to subroutine user: not using

 **** input title : coor **** incoor =  12 ****
 **** input title : elem **** incoor =  12 ****
 **** input title : stop **** incoor =  12 ****
 **** input title : cond **** inpt =  11 ****
 **** input title : ctrl **** inpt =  11 ****
 **** input title : flow **** inpt =  11 ****
 **** input title : init **** inpt =  11 ****
 **** input title : node **** inpt =  11 ****
 **** input title : perm **** inpt =  11 ****
 **** input title : rock **** inpt =  11 ****
 **** input title : sol  **** inpt =  11 ****
 **** input title : time **** inpt =  11 ****
 **** input title : mptr **** inpt =  11 ****
 mptr read from optional input file: inp.ptrk
 **** input title : stop **** inpt =  11 ****
 BC to BC connection(s) found(now set=0.0)

 pressures and temperatures set by gradients
 storage needed for ncon           2007 available       2007
 storage needed for nop            2407 available       7296
 storage needed for a matrix       6416 available       6416
 storage needed for b matrix       8016 available      29184
 storage needed for gmres          7236 available       7236
 storage available for b matrix resized to       8016<<<<<<


 time for reading input, forming coefficients   1.60



 **** analysis of input data on file fehmn.chk                        ****


 note>>tracer solution started on time step      1
 mptr read from optional input file: inp.ptrk
 Species:  1
 at time:  9.993155373032168E-02
 rest:  0.000000000000000
 Species:  2
 at time:  9.993155373032168E-02
 rest:  0.000000000000000
 Species:  3
 at time:  9.993155373032168E-02
 rest:  0.000000000000000
 Species:  4
 at time:  9.993155373032168E-02
 rest:  0.000000000000000
 Species:  5
 at time:  9.993155373032168E-02
 rest:  0.000000000000000
*********************************************************************
 Time Step         1

                    Timing Information
           Years             Days         Step Size (Days)
       0.999316E-01       0.365000E+02       0.365000E+02
 Cpu Sec for Time Step =  0.4000E-01 Current Total =    0.4000E-01

                    Equation Performance
 Number of N-R Iterations:          1
 Avg # of Linear Equation Solver Iterations:   2.0
 Number of Active Nodes:       402.
 Total Number of Iterations, N-R:          1 , Solver:          2
                    Largest Residuals
 EQ1 R=  0.5335E-13 node=   240 x=0.1900     y= 0.000     z= 1.000
 EQ2 R=  0.3113E-14 node=    54 x=0.2650     y= 1.000     z= 1.000
   Node   Equation 1 Residual   Equation 2 Residual
    201     -0.308559E-14          -0.364009E-15

                    Nodal Information (Water)
                                            source/sink   source/sink
  Node   p(MPa)   e(MJ)   l sat   temp(c)     (kg/s)        (MJ/s)
   201    1.000    0.11   1.000    25.000     0.500E-09     0.528E-10

                    Global Mass & Energy Balances
 Total mass in system at this time:            0.299168E+03 kg
 Total mass of steam in system at this time:   0.000000E+00 kg
 Total enthalpy in system at this time:        0.750546E+02 MJ

 Water discharge this time step:   0.315360E-02 kg (0.100000E-08 kg/s)
 Water input this time step:       0.315360E-02 kg (0.100000E-08 kg/s)
 Total water discharge:            0.315360E-02 kg (0.100000E-08 kg/s)
 Total water input:                0.315360E-02 kg (0.100000E-08 kg/s)

 Enthalpy discharge this time step:  0.333149E-03 MJ (0.105641E-09 MJ/s)
 Enthalpy input this time step:      0.333149E-03 MJ (0.105641E-09 MJ/s)
 Total enthalpy discharge:           0.333149E-03 MJ (0.105641E-09 MJ/s)
 Total enthalpy input:               0.333149E-03 MJ (0.105641E-09 MJ/s)

 Net kg water discharge (total out-total in):   0.464224E-11
 Net MJ discharge (total out-total in):   0.487107E-12
 Conservation Errors:   0.111155E-08 (mass),   0.139788E-09 (energy)

                   *************************
                       Particle Tracking ==> Species:   1
 Number Having Entered System:         10
 Number Currently In System  :         10
 Number Having Left System   :          0
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   2
 Number Having Entered System:          0
 Number Currently In System  :          0
 Number Having Left System   :          0
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   3
 Number Having Entered System:          0
 Number Currently In System  :          0
 Number Having Left System   :          0
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   4
 Number Having Entered System:          0
 Number Currently In System  :          0
 Number Having Left System   :          0
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   5
 Number Having Entered System:         10
 Number Currently In System  :         10
 Number Having Left System   :          0
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0
 Species:  1
```

```
 at time:  0.1002053388090349
 rest:  0.000000000000000
 Species:  2
 at time:  0.1002053388090349
 rest:  0.000000000000000
 Species:  3
 at time:  0.1002053388090349
 rest:  0.000000000000000
 Species:  4
 at time:  0.1002053388090349
 rest:  0.000000000000000
 Species:  5
 at time:  0.1002053388090349
 rest:  0.000000000000000
**********************************************************************
 Time Step        36

                    Timing Information
           Years              Days         Step Size (Days)
       0.990010E+04       0.361601E+07       0.109575E+06
 Heat and Mass Solution Disabled

                   *************************
                       Particle Tracking ==> Species:   1
 Number Having Entered System:     500000
 Number Currently In System  :          0
 Number Having Left System   :          0
 Number Having Decayed       :     500000

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   2
 Number Having Entered System:     500000
 Number Currently In System  :     433657
 Number Having Left System   :      56347
 Number Having Decayed       :       9996

  Node    Concentration    # of Particles
   201     0.410205E+04              1534

                   *************************
                       Particle Tracking ==> Species:   3
 Number Having Entered System:       9996
 Number Currently In System  :       8598
 Number Having Left System   :       1390
 Number Having Decayed       :          8

  Node    Concentration    # of Particles
   201     0.125682E+03                47

                   *************************
                       Particle Tracking ==> Species:   4
 Number Having Entered System:          8
 Number Currently In System  :          5
 Number Having Left System   :          3
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
```

```
                       Particle Tracking ==> Species:   5
 Number Having Entered System:     500000
 Number Currently In System  :     441782
 Number Having Left System   :      58218
 Number Having Decayed       :          0

  Node    Concentration    # of Particles
   201     0.215799E+04               807
 Species:  1
 at time:  10200.10403832991
 rest:  0.000000000000000
 Species:  2
 at time:  10200.10403832991
 rest:  21456.00000000000
 Species:  3
 at time:  10200.10403832991
 rest:  558.0000000000000
 Species:  4
 at time:  10200.10403832991
 rest:  2.000000000000000
 Species:  5
 at time:  10200.10403832991
 rest:  21762.00000000000
*********************************************************************
 Time Step        70

                    Timing Information
           Years              Days         Step Size (Days)
       0.199863E+05       0.730000E+07       0.680120E+05
 Heat and Mass Solution Disabled

                   *************************
                       Particle Tracking ==> Species:   1
 Number Having Entered System:     500000
 Number Currently In System  :          0
 Number Having Left System   :          0
 Number Having Decayed       :     500000

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   2
 Number Having Entered System:     500000
 Number Currently In System  :          0
 Number Having Left System   :     486844
 Number Having Decayed       :      13156

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   3
 Number Having Entered System:      13156
 Number Currently In System  :          0
 Number Having Left System   :      13082
 Number Having Decayed       :         74

  Node    Concentration    # of Particles
   201     0.000000E+00                 0

                   *************************
                       Particle Tracking ==> Species:   4
```

```
Number Having Entered System:         74
Number Currently In System  :          0
Number Having Left System   :         64
Number Having Decayed       :         10

 Node    Concentration    # of Particles
  201     0.000000E+00                 0

                  *************************
                      Particle Tracking ==> Species:   5
Number Having Entered System:     500000
Number Currently In System  :          0
Number Having Left System   :     500000
Number Having Decayed       :          0

 Node    Concentration    # of Particles
  201     0.000000E+00                 0


simulation ended: days  7.300E+06 timesteps    70


total N-R iterations =          2
total solver iterations =          4


total code time(timesteps) =   1061.690039


****---------------------------------------------------------****
**** This program for                                        ****
****   Finite Element Heat and Mass Transfer in porous media ****
****---------------------------------------------------------****
****                         Version  : FEHM V2.00 99-05-03  ****
****                         End Date : 06/14/1999           ****
****                             Time : 09:47:58             ****
****---------------------------------------------------------****
```

Routine BKPM was run to extract particle breakthrough information at the outflow boundary. The default output files from BKPM were spec_1, spec_2, spec_3, and spec_4 for speciese 1, 2, 3, and 4, respectively.

Each output file contains the following information: Time (year), mass flux (particles/year), fractions of particles left during the current time step, and normalized cummulative breakthrough data.

The example below demonstrates the use of BKPM and the sample output results.

```
picard< 41 > % bkpm
 Total number of particles:   500000
 Change the Value by Typing in a New Number or 1
1
 Calculate mass flow rate->1 or concentration->2
1
 Cal. Normalized Cumulative Breakthrough Data (Y/N)?
y

 Output Data Stored in File: spec_1
 Output Data Stored in File: spec_2
 Output Data Stored in File: spec_3
 Output Data Stored in File: spec_4
 Output Data Stored in File: spec_5

 Finished
```

BKPM extracted results for the 5 species are listed below.

File spec_1:

```
       time (year)      mass flux     fraction_left normalized

                                                 cumulative
     0.499658E-01          0.        0.000000    0.000000
     0.495010E+04          0.        0.000000    0.000000
     0.149432E+05          0.        0.000000    0.000000
```

File spec_2:

```
       time (year)      mass flux     fraction_left normalized

                                                 cumulative
     0.499658E-01          0.        0.000000    0.000000
     0.495010E+04    0.569162E+01    0.112694    0.112694
     0.149432E+05    0.426818E+02    0.860994    0.973688
```

File spec_3:

```
        time (year)           mass flux        fraction_left   normalized

                                                               cumulative
    0.499658E-01           0.           0.000000     0.000000
    0.495010E+04    0.140404E+00    0.002780     0.002780
    0.149432E+05    0.115921E+02    0.023384     0.026164
```

File spec_4:

```
        time (year)           mass flux        fraction_left   normalized

                                                               cumulative
    0.499658E-01           0.           0.000000     0.000000
    0.495010E+04    0.303030E-03    0.000006     0.000006
    0.149432E+05    0.604787E-02    0.000122     0.000128
```

File spec_5:

```
        time (year)           mass flux        fraction_left   normalized

                                                               cumulative
    0.499658E-01           0.           0.000000     0.000000
    0.495010E+04    0.588061E+01    0.116436     0.116436
    0.149432E+05    0.438006E+02    0.883564     1.000000
```

Hand calculations were performed to check the results from BKPM.

The total mass input to the system is *totmass=1* represented by a total of *500,000 (total)* particles.

The mass flow rate, fractions left, and normalized cumulative breakthrough data were calculated based on Equations (II-1), (II-2), (II-3), and (II-4), respectively.

At time t_c*=0.999316E-1 years,* t_p*=0.0 years,* t_c*-*t_p*=0.999316E-1 years*

$Species: \ i$

$$mass\_flux\_i = \frac{0-0}{0.999316\times10^{-1}} = 0, \qquad\qquad i = 1,2,3,4,5.$$

$$fractions\_left\_i = \frac{0-0}{500000} = 0, \qquad\qquad i = 1,2,3,4,5.$$

$$normalized\_cumulative\_i = \frac{0}{500000} = 0, \qquad\qquad i = 1,2,3,4,5.$$

At time t_c*=0.990010E4 years,* t_p*=0.999316E-1 years,* t_c-t_p*=0.990000E4 years*

$Species: \ 1$

$$mass\_flux\_1=\frac{0-0}{9900.00}=0$$

$$fractions\_left\_1=\frac{0}{500000}=0$$

$$normalized\_cumulative\_1=\frac{0}{500000}=0$$

$Species: \ 2$

$$mass\_flux\_2=\frac{56347-0}{9900.00}\times\frac{1}{500000}=0.113832\times10^{-4}$$

$$fractions\_left\_2=\frac{56347-0}{500000}=0.112694$$

$$normalized\_cumulative\_2=\frac{56347}{500000}=0.112694$$

$Species: \ 3$

$$mass\_flux\_3=\frac{1390-0}{9900.00}\times\frac{1}{500000}=0.280808\times10^{-6}$$

$$fractions\_left\_3=\frac{1390-0}{500000}=0.00278$$

$$normalized\_cumulative\_3=\frac{1390}{500000}=0.00278$$

$Species: \ 4$

$$mass\_flux\_4=\frac{3-0}{9900.00}\times\frac{1}{500000}=0.606061\times10^{-11}$$

$$fractions\_left\_4=\frac{3-0}{500000}=0.000006$$

$$normalized\_cumulative\_4=\frac{3}{500000}=0.000006$$

$Species: \ 5$

$$mass\_flux\_5=\frac{58218-0}{9900.00}\times\frac{1}{500000}=0.117612\times10^{-4}$$

$$fractions\_left\_5=\frac{58218-0}{500000}=0.116436$$

$$normalized\_cumulative\_5=\frac{58218}{500000}=0.116436$$

At time t_c*=0.199863E5 years*, t_p*=0.990010E4 years*, t_c-t_p*=0.100862E5 years*

$$Species: \ 1$$

$$mass\_flux\_1 = \frac{0-0}{9900.00} = 0$$

$$fractions\_left\_1 = \frac{0}{500000} = 0$$

$$normalized\_cumulative\_1 = \frac{0}{500000} = 0$$

$$Species: \ 2$$

$$mass\_flux\_2 = \frac{486844-56347}{10086.2} \times \frac{1}{500000} = 0.853636 \times 10^{-4}$$

$$fractions\_left\_2 = \frac{486844-56347}{500000} = 0.860994$$

$$normalized\_cumulative\_2 = \frac{486844}{500000} = 0.973688$$

$$Species: \ 3$$

$$mass\_flux\_3 = \frac{13082-1390}{10086.2} \times \frac{1}{500000} = 0.231842 \times 10^{-5}$$

$$fractions\_left\_3 = \frac{13082-1390}{500000} = 0.023384$$

$$normalized\_cumulative\_3 = \frac{13082}{500000} = 0.026164$$

$$Species: \ 4$$

$$mass\_flux\_4 = \frac{64-3}{10086.2} \times \frac{1}{500000} = 0.120957 \times 10^{-7}$$

$$fractions\_left\_4 = \frac{64-3}{500000} = 0.000122$$

$$normalized\_cumulative\_4 = \frac{64}{500000} = 0.000128$$

$$Species: \ 5$$

$$mass\_flux\_5 = \frac{500000-58218}{10086.2} \times \frac{1}{500000} = 0.876013 \times 10^{-4}$$

$$fractions\_left\_5 = \frac{500000-58218}{500000} = 0.883564$$

$$normalized\_cumulative\_5 = \frac{500000}{500000} = 1.0$$

Results from the hand calculations are listed in Table II-1 for comparison with the results from routine BKPM. The comparison results showed that software routine BKPM performed as designed and extracted correct results from the particle output file.

Table II-1. Comparison of BKPM and Hand Calculation Results

Time (year)	Species	Type of Cal.	Mass Flux (particle/year)	Fractions Left	Normalized Cumulative
0.999316E-1	1	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	2	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	3	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	4	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	5	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
0.99001E4	1	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	2	BKPM	5.69162	0.112694	0.112694
		hand cal.	5.69162	0.112694	0.112694
	3	BKPM	1.40404E-1	0.00278	0.00278
		hand cal.	1.40404E-1	0.00278	0.00278
	4	BKPM	3.0303E-4	0.000006	0.000006
		hand cal.	3.0303E-4	0.000006	0.000006
	5	BKPM	5.88061	0.116436	0.116436
		hand cal.	5.88061	0.116436	0.116436
0.199863E4	1	BKPM	0.0	0.0	0.0
		hand cal.	0.0	0.0	0.0
	2	BKPM	4.26818E+1	0.860994	0.97368
		hand cal.	4.26818E+1	0.860994	0.973688
	3	BKPM	1.15921	0.023384	0.026164
		hand cal.	1.15921	0.023384	0.026164
	4	BKPM	6.04787E-3	0.000122	0.000128
		hand cal.	6.04787E-3	0.000122	0.000128
	5	BKPM	4.38006E+1	0.883564	1.0
		hand cal.	4.38006E+1	0.883564	1.0

Listing of FEHM post processing software routine BKPM

```
      program bkpm

C**********************************************************************
C   bkpm is a software routine used to extract breakthrogh cureves    *
C   at the out flow boundary from the FEHM particle output file *.out *
C   The routine automatically read in FEHM control file: fehm.files   *
C   and search for the particle output file, it then open the output  *
C   and process the data to get the particle mass flow rate, the      *
C   cumulative fraction of particle flow out of the system, and the   *
C   number of particles flow out of the system at each time step.     *
C                                                                     *
C                                          Chunhong  Li, 6/14/1999     *
C**********************************************************************

      character*80 line,prx*5,cidx*5,title*31,cbyn*1
      character*60 outfile,fout*10,dfile*30,apix*10,ctmp
      real*8 time,time_old(30),reptime
      double precision conv,conc
      integer entered,cur_in,left,tot_entered,decayed
      integer left_old(30)
      character*6 spacer2
      parameter(rate_const = 3.239e-7)

      data time_old/30*0.0/left_old/30*0/


      conv=1.0
      ifehm=2
      apix='.btp'
      prx='spec_'
      title='Particle Tracking ==> Species: '
      dfile='fehmn.files'
      open(ifehm,file=dfile,status='old',iostat=iofehm)
      if(iofehm.eq.0)then
        do ip=1,4
           read(ifehm,*)outfile
        end do
      else
        write(*,*) 'Name of output file: '
        read(*,'(a60)') outfile
      endif


      call prefix(outfile,leng)


      open(7,file=outfile)

      reptime=0.

c determine number of particles in problem
 1    read(7,'(a80)',end=99) line
      if (line(16:22).eq.'Entered') read(line,72) spacer2,tot_entered
 72   format(1a30,i11)
      goto 1
 99   write(*,*) 'Total number of particles: ',tot_entered
      total=tot_entered
      write(6,*)'Change the Value by Typing in a New Number or 1'
      read(5,*)indx
      if(indx.ne.1)total=indx
      write(6,*)'Calculate mass flow rate->1 or concentration->2'
      read(5,*)indx
      if(indx.eq.2)then
      write(6,*)'Conversion factor used for concentration calculation'
      read(5,*)conv
      endif
      write(6,*)'Cal. Normalized Cumulative Breakthrough Data (Y/N)?'
      read(5,*)cbyn

      rewind(7)

      write(6,*)''
      ist=0
 2    read(7,'(a80)',end=999) line
      if (line(12:16).eq.'Years') then
        read(7,*) time
        ist=ist+1
        iout=20
        idx=0
      else
        if(line(24:54).eq.title)then
          read(line,59)ctmp,cidx
59        format(A54,a5)
          iout=iout+1
          idx=idx+1
          if(ist.eq.1)then
            do i=1,5
              if(cidx(i:i).ne. ' ')jj=i
            enddo
            fout=prx//cidx(jj:5)
            open(iout,file=fout)
            write(6,*)'Output Data Stored in File: ',fout
          endif
          read(7,'(a80)') line
          if (line(16:22).eq.'Entered') then
            read(line,72) spacer2,entered
            read(7,'(a80)') line
            read(line,72) spacer2,cur_in
            read(7,'(a80)') line
            read(line,72) spacer2,left
            read(7,'(a80)') line
            read(line,72) spacer2,decayed
            left_this = left - left_old(idx)
            delta_time = time - time_old(idx)
            if(delta_time.ne.0.)then
              reptime=time_old(idx)+0.5*delta_time
              conc = real(left_this)
              cnorm=conc/total
              conc = conv*conc/delta_time
              left_old(idx) = left
              time_old(idx) = time
              if(cbyn.eq.'y')then
                write(iout,100) reptime,conc,cnorm,left/total
              else
                write(iout,110) reptime,conc
              endif
            endif
          endif
        endif
      endif
      goto 2

 999  write(6,*)''
      write(6,*)'Finished'

100   format(1x,E13.6,3x,E13.6,2(3x,f9.6))
110   format(1x,E13.6,3x,E13.6)

      end



      subroutine prefix(fname,length)

      character*60 fname

      do length=1,60
        if(fname(length:length).eq.'.')goto 100
      end do

100   length=length-1

      return
      end
```

ATTACHMENT III. SOFTWARE ROUTINE CHAIN

Software routine CHAIN, Version 1.0, written in FORTRAN IV was acquired from U.S. Salinity Laboratory at 450 Big Springs Rd., Riverside, CA 92507. The routine was used to calculate analytical solutions for convective-dispersive transport of solutes involved in sequential first-order decay reactions. The routine version number and subsequent changes is listed below.

Routine	Version number	Changes made
CHAIN	May 1982. Code issuer did not have a version number.	None

The theory and applications of CHAIN were published by M. Th. van Genuchten (van Genuchten 1985). The paper documented in detail the theory, assumptions, and boundary conditions used in deriving the analytical solutions and the code CHAIN developed for calculating the analytical solutions. Two examples on the applications of CHAIN were given in the paper.

The executable version of CHAIN was installed on a DELL OptiPlex GX1 PC with windows NT 4.0 operating system at Sandia National Laboratories in Albuquerque, New Mexico. The PC has 128 MB memory, 9 GB hard disk drive, and a CPU speed of 300 MHz. No modification was made to CHAIN after it was received or installed.

Hand calculations were carried out to check against the results from CHAIN. A 4 species chain decay model in a semi-infinite system with 0 initial concentrations was used in the comparison. Only species 1 was released at x=0 from time 0 to t_0 years at a constant concentration of B_1.

$$
\begin{aligned}
&c_1(0,t) = \begin{cases} B_1, & \qquad\qquad 0 \prec t \prec t_0 \\ 0, & \qquad\qquad t \succ t_0 \end{cases} \\
&c_i(0,t) = 0, \qquad\qquad\qquad\qquad i \geq 2 \\
&\frac{\partial c_i}{\partial x}(\infty, t) = 0, \qquad\qquad\qquad t \geq 0(i = 1,4)
\end{aligned}
$$

The solutions for this system are (van Genuchten, 1985):

$$
c_i = \begin{cases} c_i^*(x,t), & 0 \prec t \leq t_0 \\ c_i^*(x,t) - \exp(-\lambda_i t_0) c_i^*(x, t - t_0), & t \succ t_0 \end{cases} \qquad \text{(Eq. III-1)}
$$

where $\lambda_i = N_b \mu_i + \gamma_i$, μ_i is the rate constanstant of species i for first order decay, γ_i is the source term release rate for species i, and N_b equals 1 or 0. For $N_b=1$, Bateman equations are used for input boundary conditions, for $N_b=0$, Bateman equations are not used, parameter B_i are read in, x is the distance from the origin of source release, and c_i^* is part of the analytical solution for species i and was given in appendix 1 in van Genuchten's paper (van Genuchten, 1985) with

$$
\begin{aligned}
c_1^* &= B_1 F_{110} \\
c_2^* &= \frac{\mu_1 R_1 B_1 \left(S_{12} - F_{210} + F_{110}\right)}{R_{12}\lambda_1 - \mu_{12}} \\
c_3^* &= \frac{A_1 B_1 R_{12}\left(F_{210} - F_{110} - S_{12}\right)}{R_{12}\lambda_1 - \mu_{12}} + \frac{A_1 B_1 R_{13}\left(S_{13} - F_{310} + F_{110}\right)}{R_{13}\lambda_1 - \mu_{13}} + \frac{A_1 B_1 R_{23}\left(F_{310} - F_{210} - S_{23}\right)}{R_{23}\lambda_1 - \mu_{23}} \\
c_4^* &= \frac{A_3 B_1 R_{14}^2\left(S_{14} - F_{410} + F_{110}\right)}{G_{134}G_{124}(R_{14}\lambda_1 - \mu_{14})} + \frac{A_3 B_1 R_{24}^2\left(F_{410} - F_{210} - S_{24}\right)}{G_{124}G_{234}(R_{24}\lambda_1 - \mu_{24})} + \frac{A_3 B_1 R_{34}^2\left(S_{34} - F_{410} + F_{310}\right)}{G_{134}G_{234}(R_{34}\lambda_1 - \mu_{34})} + \\
&\quad \frac{A_3 B_1 R_{13}^2\left(F_{310} - F_{110} - S_{13}\right)}{G_{123}G_{134}(R_{13}\lambda_1 - \mu_{13})} + \frac{A_3 B_1 R_{23}^2\left(S_{23} - F_{310} + F_{210}\right)}{G_{123}G_{234}(R_{23}\lambda_1 - \mu_{23})} + \frac{A_3 B_1 R_{12}^2\left(S_{12} - F_{210} + F_{110}\right)}{G_{123}G_{124}(R_{12}\lambda_1 - \mu_{12})}
\end{aligned}
\qquad \text{(Eq. III-2)}
$$

With

$$
\begin{aligned}
R_{ij} &= R_i - R_j \\
\mu_{ij} &= \mu_i - \mu_j \\
G_{ijk} &= \mu_i R_i R_{kj} + \mu_j R_j R_{ik} + \mu_k R_k R_{ji} \\
A_1 &= \frac{\mu_1 \mu_2 R_1 R_2}{G_{123}} \\
A_2 &= \frac{\mu_2 \mu_3 R_2 R_3}{G_{234}} \\
A_3 &= \mu_1 \mu_2 \mu_3 R_1 R_2 R_3
\end{aligned}
$$

$$
F_{ijk} = \exp\left(-a_{ijk}t\right)\left\{\frac{1}{2}\exp\left[\frac{(v-w)x}{2D}\right]erfc\left[\frac{R_i x - wt}{2\left(DR_i t\right)^{1/2}}\right] + \frac{1}{2}\exp\left[\frac{(v+w)x}{2D}\right]erfc\left[\frac{R_i x + wt}{2\left(DR_i t\right)^{1/2}}\right]\right\} \qquad \text{(Eq. III-3)}
$$

$$
S_{ij} = F_{jji} - F_{iji}
$$

wehre

$$
a_{ijk} = \begin{cases} \lambda_j, & k = 0 \\ \dfrac{\mu_{ij}}{R_{ij}}, & k > 0 \end{cases}
$$

$$
w = \left[v^2 + 4DR_i\left(\mu_i - a_{ijk}\right)\right]^{\frac{1}{2}}
$$

In the above equations, v is average pore water velocity, and R_i is the ratardation factor of the i th species.

Hand calculations were carried out for a 4 member sequential decay chain system to check the correctness of the CHIAN results under the same conditions. The half lives for species 1 through 4 were 1,000, 3,000, 20,000, and 4,000 years, respectively. Species 1 was released at x=0 from time 0 to 5,000 years at constant concentration of B_1=1.0. Solute concentrations of each species were calculated at a distance of $x=1.0$ m. Other parameters used in the calculations are listed below.

$N_b=0$, $B_1=1$, $B_2= B_3=B_4=0$, $\gamma_i=0$ $(i=1,4)$, $x=1.0$ *m*, $\lambda_i=N_b\mu_i+\gamma_{i,}=0$ $(i=1,2,3,$*and* $4)$
$\mu_1=6.9315E\text{-}4/year$, $\mu_2=2.3105E\text{-}4/year$, $\mu_3=3.4657E\text{-}5/year$, $\mu_4=1.7329E\text{-}4/year$, $v=1.0519E\text{-}4$ *m/year*, $D=5.2595E\text{-}7$ $m^2/year$, $R_1=1.0$, $R_2=1.0$, $R_3=1.001$, and $R_4=1.001$.

Software routine CHAIN was used to calculate the solutes breakthrough curves at *x=1.0* m at a time interval of 300 years from 0 to 20,000 years. The CHAIN input and output files are attached with this document for referencing. Hand calculations were then carried out to randomly check the results from CHAIN.

The following parameters were calculated based on the input parameters given above.

R_{12}=0, R_{13}=-1.E-3, R_{14}=-1.E-3, R_{23}=-1.E-3, R_{24}=-1.E-3, R_{34}=0

$\mu_{12}=4.6210E\text{-}4$, $\mu_{13}=6.5849E\text{-}4$, $\mu_{14}=5.1986E\text{-}4$, $\mu_{23}=1.9639E\text{-}4$, $\mu_{24}=0.5776E\text{-}4$, $\mu_{34}=-1.3863E\text{-}4$

$G_{123}=4.6212E\text{-}7$ $G_{124}=4.6212E\text{-}7$ $G_{134}=-1.3878E\text{-}7$ $G_{234}=-1.3878E\text{-}7$

$A_1=0.3466$ $A_3=5.5560E\text{-}12$

The partial solutions c_i^* can be rewritten as

$$
\begin{aligned}
c_1^* &= B_1 F_{110} \\
c_2^* &= C_{21}\left(S_{12} - F_{210} + F_{110}\right) \\
c_3^* &= C_{31}(F_{210} - F_{110} - S_{12}) + C_{32}\left(S_{13} - F_{310} + F_{110}\right) + C_{33}\left(F_{310} - F_{210} - S_{23}\right) \\
c_4^* &= C_{41}\left(S_{14} - F_{410} + F_{110}\right) + C_{42}\left(F_{410} - F_{210} - S_{24}\right) + C_{43}\left(S_{34} - F_{410} + F_{310}\right) + \\
&\quad C_{44}\left(F_{310} - F_{110} - S_{13}\right) + C_{45}\left(S_{23} - F_{310} + F_{210}\right) + C_{46}\left(S_{12} - F_{210} + F_{110}\right)
\end{aligned}
\qquad \text{(Eq. III-4)}
$$

With

$$C_{21} = \frac{\mu_1 R_1 B}{R_{12}\lambda_1 - \mu_{12}} = -1.5000 \qquad C_{31} = \frac{A_1 B_1 R_{12}}{R_{12}\lambda_1 - \mu_{12}} = 0.0000$$

$$C_{32} = \frac{A_1 B_1 R_{13}}{R_{13}\lambda_1 - \mu_{13}} = 0.5264 \qquad C_{33} = \frac{A_1 B_1 R_{23}}{R_{23}\lambda_1 - \mu_{23}} = 1.7649$$

$$C_{41} = \frac{A_3 B_1 R_{14}^2}{G_{134} G_{124} (R_{14}\lambda_1 - \mu_{14})} = -0.1666 \qquad C_{42} = \frac{A_3 B_1 R_{24}^2}{G_{124} G_{234} (R_{24}\lambda_1 - \mu_{24})} = -1.5000$$

$$C_{43} = \frac{A_3 B_1 R_{34}^2}{G_{134} G_{234} (R_{34}\lambda_1 - \mu_{34})} = 0.0000 \qquad C_{44} = \frac{A_3 B_1 R_{13}^2}{G_{123} G_{134} (R_{13}\lambda_1 - \mu_{13})} = -0.1316$$

$$C_{45} = \frac{A_3 B_1 R_{23}^2}{G_{123} G_{234} (R_{23}\lambda_1 - \mu_{23})} = -0.4412 \qquad C_{46} = \frac{A_3 B_1 R_{12}^2}{G_{123} G_{124} (R_{12}\lambda_1 - \mu_{12})} = 0.0000$$

To compare against the solutions from routine CHAIN at *x=1.0 m*, we needed to calculate functions F_{ijk} at a given time *t* and solve for c_i. In van Genutchen's paper, approximation was made to calculate the products of *exp(A)erfc(B)* in F_{ijk} using the following formula.

For $0 \le B \le 2.5$

$$\exp(A) erfc(B) \approx \exp(A - B^2)(\alpha_1\tau + \alpha_2\tau^2 + \alpha_3\tau^3 + \alpha_4\tau^4 + \alpha_5\tau^5)$$

where

$$\tau = \frac{1}{1 + 0.3275911B}, \qquad \alpha_1 = 0.254829592$$
$$\alpha_2 = -0.284496736, \qquad \alpha_3 = 1.421413741$$
$$\alpha_4 = -1.453152027, \qquad \alpha_5 = 1.061405429$$

For $B > 2.5$

$$\exp(A) erfc(B) \approx \frac{1}{\sqrt{\pi}} \frac{\exp(A - B^2)}{(B + 0.5/(B + 1/(B + 1.5/(B + 2/(B + 2.5/(B + 1))))))}$$

For $B < 0$

$$\exp(A) erfc(B) = 2\exp(A) - \exp(A) erfc(-B)$$

and

$$\exp(A) erfc(B) \approx 0 \quad \mathit{for} \quad \begin{cases} |A| > M, \\ B \le 0, \end{cases} \quad \mathit{or} \quad \begin{cases} |A - B^2| > M, \\ B > 0, \end{cases} \quad (M = 170).$$

and S_{ij}*=0,* for *abs(1000xR*$_{ij}$*/R*$_i$*)<1* (van Genutchen, 1985)

Since in our case, the absolute ratio of *abs(*1000xR$_{ij}$*/R*$_i$*)<1*, S_{ij}*=0* in the calculations.

Hand calculations were carried out to check the results of CHAIN at selected time of *t=3000, 8100, 12300,* and *17400* years.

At time *t=3000* years:

Substituting parameters into the Equation (3), we calculated functions F_{ijk} at the specified time.

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6643}{0.0795}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3357}{0.0794})\right] = 3.6631\times10^{-35}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6776}{0.0795}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3224}{0.0794})\right] = 1.4512\times10^{-34}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6844}{0.0795}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3176}{0.0795})\right] = 2.2572\times10^{-34}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6803}{0.0795}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3217}{0.0795})\right] = 1.4935\times10^{-34}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

As *t<5,000 years,* $c_i = c_i^*$, From Equation (1), the normalized solute concentrations at *t=3,000 years* are

$$\begin{aligned}
c_1 &= B_1 F_{110} = 1\times3.6631\times10^{-35}\\
&= 3.6631\times10^{-35}\\
c_2 &= C_{21}(F_{110} - F_{210}) = -1.5(3.6631\times10^{-35} - 1.4512\times10^{-34})\\
&= 1.6273\times10^{-34}\\
c_3 &= C_{32}(F_{110} - F_{310}) + C_{33}(F_{310} - F_{210}) = 0.5264(3.6631\times10^{-35} - 2.2572\times10^{-34}) +\\
&\quad 1.7649(2.2572\times10^{-34} - 1.4512\times10^{-34}) = -0.5264\times1.8909\times10^{-34} + 1.7649\times8.06\times10^{-34}\\
&= 4.2714\times10^{-35}\\
c_4 &= C_{41}(F_{110} - F_{410}) + C_{42}(F_{410} - F_{210}) + C_{44}(F_{310} - F_{110}) + C_{45}(F_{210} - F_{310})\\
&= -0.1666(3.6631\times10^{-35} - 1.4935\times10^{-34}) - 1.5(1.4935\times10^{-34} - 1.4512\times10^{-34}) -\\
&\quad 0.1316(2.2572\times10^{-34} - 3.6631\times10^{-35}) - 0.4412(1.4512\times10^{-34} - 2.2572\times10^{-34})\\
&= 2.3111\times10^{-35}
\end{aligned}$$

At time *t=8100* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.0936}{0.1305}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.9065}{0.1305})\right] = 2.8140\times10^{-4}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.1294}{0.1305}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.8706}{0.1305})\right] = 9.9902\times10^{-3}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.1462}{0.1306}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.8558}{0.1306})\right] = 4.4908\times10^{-2}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.1350}{0.1306}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.8670}{0.1306})\right] = 1.5318\times10^{-2}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

For *(t-t$_0$)=8100-5000=3100* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6531}{0.0808}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3469}{0.0808})\right] = 3.4337\times10^{-33}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6668}{0.0808}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3332}{0.0808})\right] = 1.4237\times10^{-32}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6738}{0.0808}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3282}{0.0808})\right] = 2.2703\times10^{-32}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.6696}{0.0808}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.3324}{0.0808})\right] = 1.4818\times10^{-32}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

From Equation (III-1), the normalized solute concentrations at *t=8,100 years* are

$$
\begin{aligned}
c_1 &= c_1^*(1,8100) - c_1^*(1,3100) == 2.8140\times 10^{-4} - 3.4337\times 10^{-33} \\
&= 2.8140\times 10^{-4} \\
c_2 &= c_2^*(1,8100) - c_2^*(1,3100) = \left[-1.5(2.8140\times 10^{-4} - 9.9902\times 10^{-3})\right] - \\
&\quad \left[-1.5(3.4337\times 10^{-33} - 1.4237\times 10^{-32})\right] \\
&= 1.4563\times 10^{-2} \\
c_3 &= c_3^*(1,8100) - c_3^*(1,3100) = [0.5264(2.8140\times 10^{-4} - 4.4908\times 10^{-2}) + \\
&\quad 1.7649(4.4908\times 10^{-2} - 9.9902\times 10^{-3})] - [0.5264(4.4337\times 10^{-33} - 2.2703\times 10^{-32}) + \\
&\quad 1.7649(2.2703\times 10^{-32} - 1.4237\times 10^{-32})] \\
&= 3.8135\times 10^{-2} \\
c_4 &= c_4^*(1,8100) - c_4^*(1,3100) = [-0.1666(2.8140\times 10^{-4} - 1.5318\times 10^{-2}) - \\
&\quad 1.5(1.5318\times 10^{-2} - 9.9902\times 10^{-3}) - 0.1316(4.4908\times 10^{-2} - 2.8140\times 10^{-4}) - \\
&\quad 0.4412(9.9903\times 10^{-3} - 4.4908\times 10^{-2})] - [-0.1666(3.4337\times 10^{-33} - 1.4818\times 10^{-32}) - \\
&\quad 1.5(1.4818\times 10^{-2} - 1.4237\times 10^{-32}) - 0.1316(2.2703\times 10^{-32} - 3.4337\times 10^{-33}) - \\
&\quad 0.4412(1.4237\times 10^{-32} - 2.2703\times 10^{-32})] \\
&= 4.0463\times 10^{-3}
\end{aligned}
$$

At time *t=12300* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3765}{0.1609}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3765}{0.1609})\right] = 1.6850\times10^{-3}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3220}{0.1609}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3220}{0.1609})\right] = 1.1362\times10^{-1}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.2971}{0.1609}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.2991}{0.1609})\right] = 7.1668\times10^{-1}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3140}{0.1609}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3160}{0.1609})\right] = 1.9434\times10^{-1}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

For *(t-t$_0$)=12300-5000=7300* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.1831}{0.1239}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.8169}{0.1239})\right] = 3.4569\times10^{-5}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.2154}{0.1239}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.7846}{0.1239})\right] = 9.0281\times10^{-4}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.2306}{0.1240}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.7714}{0.1240})\right] = 3.5172\times10^{-3}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{0.2206}{0.1240}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{1.7815}{0.1240})\right] = 1.3187\times10^{-3}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

From Equation (III-1), the normalized solute concentrations at *t=12,300 years* are

$$c_1 = c_1^*(1,12300) - c_1^*(1,7300) == 1.6850\times10^{-3} - 3.4569\times10^{-5}$$

$$= 1.6504\times10^{-3}$$

$$c_2 = c_2^*(1,12300) - c_2^*(1,7300) = \left[-1.5(1.6850\times10^{-3} - 1.1362\times10^{-1})\right] -$$

$$\left[-1.5(3.4569\times10^{-5} - 9.0281\times10^{-4})\right]$$

$$= 1.6660\times10^{-1}$$

$$c_3 = c_3^*(1,12300) - c_3^*(1,7300) = [0.5264(1.6850\times10^{-3} - 7.1668\times10^{-1}) +$$

$$1.7649(7.1668\times10^{-1} - 1.1362\times10^{-1})] - [0.5264(3.4569\times10^{-5} - 3.5172\times10^{-3}) +$$

$$1.7649(3.5172\times10^{-3} - 9.0281\times10^{-4})]$$

$$= 6.8338\times10^{-1}$$

$$c_4 = c_4^*(1,12300) - c_4^*(1,7300) = [-0.1666(1.6850\times10^{-3} - 1.9434\times10^{-1}) -$$

$$1.5(1.9434\times10^{-1} - 1.1362\times10^{-1}) - 0.1316(7.1668\times10^{-1} - 1.6850\times10^{-3}) -$$

$$0.4412(1.1362\times10^{-1} - 7.1668\times10^{-1})] - [-0.1666(3.4569\times10^{-5} - 1.3187\times10^{-3}) -$$

$$1.5(1.3187\times10^{-3} - 9.0281\times10^{-4}) - 0.1316(3.5172\times10^{-3} - 3.4569\times10^{-5}) -$$

$$0.4412(9.0281\times10^{-4} - 3.5172\times10^{-3})]$$

$$= 8.2658\times10^{-2}$$

At time *t=17400* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.9472}{0.1913}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.9472}{0.1913})\right] = 1.6856\times10^{-3}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.8701}{0.1913}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.8701}{0.1913})\right] = 1.1385\times10^{-1}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.8353}{0.1914}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.8373}{0.1914})\right] = 7.1946\times10^{-1}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.8592}{0.1914}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.8612}{0.1914})\right] = 1.9482\times10^{-1}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

For *(t-t$_0$)=17400-5000=12400* years

$$F_{110} = 0.5\left[\exp(\frac{-6.7170\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3876}{0.1615}) + \exp(\frac{2.1710\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3877}{0.1615})\right] = 1.6851\times10^{-3}$$

$$F_{210} = 0.5\left[\exp(\frac{-2.2857\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3327}{0.1615}) + \exp(\frac{2.1267\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3327}{0.1615})\right] = 1.1368\times10^{-1}$$

$$F_{310} = 0.5\left[\exp(\frac{-3.4634\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3077}{0.1616}) + \exp(\frac{2.1073\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3097}{0.1616})\right] = 7.1729\times10^{-1}$$

$$F_{410} = 0.5\left[\exp(\frac{-1.7206\times10^{-6}}{1.0519\times10^{-6}})erfc(\frac{-0.3247}{0.1616}) + \exp(\frac{2.1210\times10^{-4}}{1.0519\times10^{-6}})erfc(\frac{2.3267}{0.1616})\right] = 1.9445\times10^{-1}$$

$$S_{12} = S_{13} = S_{23} = S_{14} = S_{24} = S_{34} = 0$$

From Equation (III-1), the normalized solute concentrations at *t=17,400 years* are

$$
\begin{aligned}
c_1 &= c_1^*(1{,}17400) - c_1^*(1{,}12400) == 1.6856\times10^{-3} - 1.6851\times10^{-3} \\
&= 5.0\times10^{-7} \\
c_2 &= c_2^*(1{,}17400) - c_2^*(1{,}12400) = \left[-1.5(1.6856\times10^{-3} - 1.1385\times10^{-1})\right] - \\
&\left[-1.5(1.6851\times10^{-3} - 1.1368\times10^{-1})\right] \\
&= 2.5425\times10^{-4} \\
c_3 &= c_3^*(1{,}17400) - c_3^*(1{,}12400) = [0.5264(1.6856\times10^{-3} - 7.1946\times10^{-1}) + \\
&1.7649(7.1946\times10^{-1} - 1.1385\times10^{-1})] - [0.5264(1.6851\times10^{-3} - 7.1729\times10^{-1}) + \\
&1.7649(7.1729\times10^{-1} - 1.1368\times10^{-1})] \\
&= 2.3878\times10^{-3} \\
c_4 &= c_4^*(1{,}17400) - c_4^*(1{,}12400) = [-0.1666(1.6856\times10^{-3} - 1.9482\times10^{-1}) - \\
&1.5(1.9482\times10^{-1} - 1.1385\times10^{-1}) - 0.1316(7.1946\times10^{-1} - 1.6856\times10^{-3}) - \\
&0.4412(1.1385\times10^{-1} - 7.1946\times10^{-1})] - [-0.1666(1.6851\times10^{-3} - 1.9445\times10^{-1}) - \\
&1.5(1.9445\times10^{-1} - 1.1368\times10^{-1}) - 0.1316(7.1729\times10^{-1} - 1.6851\times10^{-3}) - \\
&0.4412(1.1368\times10^{-1} - 7.1729\times10^{-1})] \\
&= 3.5845\times10^{-4}
\end{aligned}
$$

Table III-1. Comparison of CHAIN and Hand Calculation Results

Time (year)	Species 1		Species 2		Species 3		Species 4	
	Hand Calculation	CHAIN	Hand Calculation	CHAIN	Hand Calculation	CHAIN	Hand Calculation	CHAIN
3000	3.6631E-35	3.663E-35	1.6273E-34	1.627E-34	4.2714E-35	4.2703E-34	2.3111E-35	2.312E-34
81000	2.8140E-4	2.814E-4	1.4563E-2	1.456E-2	3.8135E-2	3.814E-2	4.0463E-3	4.026E-3
12300	1.6504E-3	1.650E-3	1.6660E-1	1.666E-1	6.8338E-1	6.853E-1	8.2658E-2	8.240E-2
174000	5.000E-7	4.791E-7	2.5425E-4	2.571E-4	2.3878E-3	2.390E-3	3.5845E-4	3.608E-4

The hand calculation results compared well against the results from the software routine CHAIN. The small difference was due to truncation errors in hand calculations, as software routine CHAIN used double precision in the code. The comparison proved that CHAIN was correct.

The analytical solution formulated in CHAIN had certain limitations on the range of parameters. Those limitations are listed below.

(1) The number of species in the problem must be less or equal to 4.

(2) The difference of $(R_{ij}\lambda_k\text{-}\mu_{ij})$ does not equal 0. Where $R_{ij}=R_i\text{-}R_j$ and $\mu_{ij}=\mu_i R_i\text{-}\mu_j R_j$ with *i, j*=1, 2, 3, and 4, but *i* does not equal *j*, and *k*=1, 2, and 3. This constraint prevents some of the parameters from being divided by 0.

ATTACHMENT IV. SOFTWARE ROUTINE TEST_EBS_RANDOM

The test routine, *test_ebs_random.f* Version 1.0, was written in Fortran 90. The core of the test routine is the EBS random release model with added code for generating random node locations within a given field, loading nodes into an array containing node coordinates, and randomly assigning each node an index. This attachment documents that the software routine performs as expected.

LIST OF INPUTS AND SCREEN OUTPUTS OF THE TEST PROBLEM

Listed below are the screen input and output of the test problem.

```
Input total Number of Nodes
40
Input number of sub-regions
2
Input the seed for the random number geneator
13131
Input total number of M_fine failed packages
6
Input number of species
1
Released at this test (->1) or (->0)
1
For sub-region:            1
Input xmin and xmax
0 100
Input ymin and ymax
0 100
Input number of nodes in zone           1
20
How many M_fine failed package in this region
3
For sub-region:            2
Input xmin and xmax
120 250
Input ymin and ymax
120 250
Input number of nodes in zone           2
20
How many M_fine failed package in this regio
3
Node coordinates generated
Input how many time steps to simulate
4
Time step:            1
Input data for zone            1
Input number of N_failed package
10
Input data for zone            2
Input number of N_failed package
10
Time step:            2
Input data for zone            1
Input number of N_failed package
15
Input data for zone            2
Input number of N_failed package
15
Time step:            3
Input data for zone            1
Input number of N_failed package
17
```

```
Input data for zone            2
Input number of N_failed package
17
Time step:            4
Input data for zone            1
Input number of N_failed package
20
Input data for zone            2
Input number of N_failed package
20
```

LIST OF OUTPUT RESULTS CONTAINED IN FILE NODE

```
     40                                 !total_nodes, total number of nodes
      2                                 !N_large, number of sub-regions
  13131                                 !seed, for random number generator
      6                                 !M_fine, number of M_fine failed packages
      1                                 !nspecies, number of species
      1                                 !irelease, Released at this test (->1 or ->0)


!Input parameters for sub-region:       1

(      0.00,    100.00)                 !xmin, xmax, X-range
(      0.00,    100.00)                 !ymin, ymax, Y-range

     20                                 !number of nodes in zone      1

      8        92.76        97.69       !node_index,(x,y)
      3        58.12        70.45       !node_index,(x,y)
     11        14.59        94.89       !node_index,(x,y)
     13        78.99        58.85       !node_index,(x,y)
     19        11.36        27.01       !node_index,(x,y)
      1        30.52        80.25       !node_index,(x,y)
      7        56.41        46.25       !node_index,(x,y)
     24        35.86        40.80       !node_index,(x,y)
     23        37.75        59.77       !node_index,(x,y)
     20        40.77        74.52       !node_index,(x,y)
     10         7.88        97.37       !node_index,(x,y)
      5        98.07         5.48       !node_index,(x,y)
     37        36.23         3.17       !node_index,(x,y)
      6        44.49        44.11       !node_index,(x,y)
     35        85.75        43.51       !node_index,(x,y)
     16         3.88        54.84       !node_index,(x,y)
     14        66.48        84.95       !node_index,(x,y)
     18        11.24        99.72       !node_index,(x,y)
     25        19.51        10.98       !node_index,(x,y)
     15        22.46        11.69       !node_index,(x,y)

      3                                 !number of individually failed package
     93.58        48.74                 !(x,y) of failed package
     34.24         3.88                 !(x,y) of failed package
     24.39        75.84                 !(x,y) of failed package

!Input parameters for sub-region:       2

(    120.00,    250.00)                 !xmin, xmax, X-range
(    120.00,    250.00)                 !ymin, ymax, Y-range

     20                                 !number of nodes in zone      2

     32       235.34       236.86       !node_index,(x,y)
     40       128.89       174.41       !node_index,(x,y)
      4       209.61       231.01       !node_index,(x,y)
     29       213.41       182.47       !node_index,(x,y)
     31       186.02       120.08       !node_index,(x,y)
      9       190.41       188.59       !node_index,(x,y)
     30       209.18       241.93       !node_index,(x,y)
     39       223.25       163.77       !node_index,(x,y)
     22       174.71       206.49       !node_index,(x,y)
     27       136.03       219.27       !node_index,(x,y)
     12       172.61       156.08       !node_index,(x,y)
     26       136.88       147.70       !node_index,(x,y)
     38       133.46       193.26       !node_index,(x,y)
     33       181.27       144.80       !node_index,(x,y)
     21       151.83       218.28       !node_index,(x,y)
     17       186.94       184.39       !node_index,(x,y)
      2       134.16       124.11       !node_index,(x,y)
     34       196.44       167.73       !node_index,(x,y)
     36       217.48       226.39       !node_index,(x,y)
     28       129.11       158.84       !node_index,(x,y)

      3                                 !number of individually failed package
    205.62       217.01                 !(x,y) of failed package
    147.41       152.02                 !(x,y) of failed package
    135.88       207.18                 !(x,y) of failed package

Node coordinates generated

      4                                 !nsteps, number of time steps


At time step:    1                      !istp, time step

     10                                 !Number of failed packages in sub-region:      1

     10                                 !Number of failed packages in sub-region:      2

!locating node having least dist. to the given (x,y)

     93.58     48.74     35         1   !(x,y),node index,sub-region
     34.24      3.88     37         1   !(x,y),node index,sub-region
     24.39     75.84      1         1   !(x,y),node index,sub-region
    205.62    217.01      4         2   !(x,y),node index,sub-region
    147.41    152.02     26         2   !(x,y),node index,sub-region
    135.88    207.18     27         2   !(x,y),node index,sub-region

      3   !Fine packages failed in N_large zone:      1

     35      -1.00                      !node index,pcnsk
     37      -1.00                      !node index,pcnsk
      1      -1.00                      !node index,pcnsk

     10   !Packages failed in N_large zone:           1

     25      -1.00                      !node index,pcnsk
     24      -1.00                      !node index,pcnsk
     20      -1.00                      !node index,pcnsk
     23      -1.00                      !node index,pcnsk
     15      -1.00                      !node index,pcnsk
     10      -1.00                      !node index,pcnsk
     13      -1.00                      !node index,pcnsk
     19      -1.00                      !node index,pcnsk
     18      -1.00                      !node index,pcnsk
      7      -1.00                      !node index,pcnsk

      3   !Fine packages failed in N_large zone:      2

      4      -1.00                      !node index,pcnsk
     26      -1.00                      !node index,pcnsk
     27      -1.00                      !node index,pcnsk

     10   !Packages failed in N_large zone:           2

     31      -1.00                      !node index,pcnsk
     30      -1.00                      !node index,pcnsk
     29      -1.00                      !node index,pcnsk
      9      -1.00                      !node index,pcnsk
     34      -1.00                      !node index,pcnsk
     39      -1.00                      !node index,pcnsk
     12      -1.00                      !node index,pcnsk
     32      -1.00                      !node index,pcnsk
     28      -1.00                      !node index,pcnsk
     17      -1.00                      !node index,pcnsk

At time step:    2                      !istp, time step

     15                                 !Number of failed packages in sub-region:      1

     15                                 !Number of failed packages in sub-region:      2
```

```
!locating node having least dist. to the given (x,y)


      3   !Fine packages failed in N_large zone:      1

     35      -1.00                      !node index,pcnsk
     37      -1.00                      !node index,pcnsk
      1      -1.00                      !node index,pcnsk

     15   !Packages failed in N_large zone:           1

     25      -1.00                      !node index,pcnsk
     24      -1.00                      !node index,pcnsk
     20      -1.00                      !node index,pcnsk
     23      -1.00                      !node index,pcnsk
     15      -1.00                      !node index,pcnsk
     10      -1.00                      !node index,pcnsk
     13      -1.00                      !node index,pcnsk
     19      -1.00                      !node index,pcnsk
     18      -1.00                      !node index,pcnsk
      7      -1.00                      !node index,pcnsk
      6      -1.00                      !node index,pcnsk
     16      -1.00                      !node index,pcnsk
      5      -1.00                      !node index,pcnsk
     14      -1.00                      !node index,pcnsk
      8      -1.00                      !node index,pcnsk

      3   !Fine packages failed in N_large zone:      2

      4      -1.00                      !node index,pcnsk
     26      -1.00                      !node index,pcnsk
     27      -1.00                      !node index,pcnsk

     15   !Packages failed in N_large zone:           2

     31      -1.00                      !node index,pcnsk
     30      -1.00                      !node index,pcnsk
     29      -1.00                      !node index,pcnsk
      9      -1.00                      !node index,pcnsk
     34      -1.00                      !node index,pcnsk
     39      -1.00                      !node index,pcnsk
     12      -1.00                      !node index,pcnsk
     32      -1.00                      !node index,pcnsk
     28      -1.00                      !node index,pcnsk
     17      -1.00                      !node index,pcnsk
     36      -1.00                      !node index,pcnsk
     40      -1.00                      !node index,pcnsk
     22      -1.00                      !node index,pcnsk
     38      -1.00                      !node index,pcnsk
     33      -1.00                      !node index,pcnsk

At time step:    3                      !istp, time step

     17                                 !Number of failed packages in sub-region:      1

     17                                 !Number of failed packages in sub-region:      2

!locating node having least dist. to the given (x,y)


      3   !Fine packages failed in N_large zone:      1

     35      -1.00                      !node index,pcnsk
     37      -1.00                      !node index,pcnsk
      1      -1.00                      !node index,pcnsk

     17   !Packages failed in N_large zone:           1

     25      -1.00                      !node index,pcnsk
     24      -1.00                      !node index,pcnsk
     20      -1.00                      !node index,pcnsk
     23      -1.00                      !node index,pcnsk
     15      -1.00                      !node index,pcnsk
     10      -1.00                      !node index,pcnsk
     13      -1.00                      !node index,pcnsk
     19      -1.00                      !node index,pcnsk
     18      -1.00                      !node index,pcnsk
      7      -1.00                      !node index,pcnsk
      6      -1.00                      !node index,pcnsk
     16      -1.00                      !node index,pcnsk
      5      -1.00                      !node index,pcnsk
     14      -1.00                      !node index,pcnsk
      8      -1.00                      !node index,pcnsk
      3      -1.00                      !node index,pcnsk
     11      -1.00                      !node index,pcnsk

      3   !Fine packages failed in N_large zone:      2

      4      -1.00                      !node index,pcnsk
     26      -1.00                      !node index,pcnsk
     27      -1.00                      !node index,pcnsk

     17   !Packages failed in N_large zone:           2

     31      -1.00                      !node index,pcnsk
     30      -1.00                      !node index,pcnsk
     29      -1.00                      !node index,pcnsk
      9      -1.00                      !node index,pcnsk
     34      -1.00                      !node index,pcnsk
     39      -1.00                      !node index,pcnsk
     12      -1.00                      !node index,pcnsk
     32      -1.00                      !node index,pcnsk
     28      -1.00                      !node index,pcnsk
     17      -1.00                      !node index,pcnsk
     36      -1.00                      !node index,pcnsk
     40      -1.00                      !node index,pcnsk
     22      -1.00                      !node index,pcnsk
     38      -1.00                      !node index,pcnsk
     33      -1.00                      !node index,pcnsk
     21      -1.00                      !node index,pcnsk
      2      -1.00                      !node index,pcnsk

At time step:    4                      !istp, time step

     20                                 !Number of failed packages in sub-region:      1

     20                                 !Number of failed packages in sub-region:      2

!locating node having least dist. to the given (x,y)


      3   !Fine packages failed in N_large zone:      1

     35      -1.00                      !node index,pcnsk
     37      -1.00                      !node index,pcnsk
      1      -1.00                      !node index,pcnsk

     17   !Packages failed in N_large zone:           1

     25      -1.00                      !node index,pcnsk
     24      -1.00                      !node index,pcnsk
     20      -1.00                      !node index,pcnsk
     23      -1.00                      !node index,pcnsk
     15      -1.00                      !node index,pcnsk
     10      -1.00                      !node index,pcnsk
     13      -1.00                      !node index,pcnsk
     19      -1.00                      !node index,pcnsk
     18      -1.00                      !node index,pcnsk
      7      -1.00                      !node index,pcnsk
      6      -1.00                      !node index,pcnsk
     16      -1.00                      !node index,pcnsk
      5      -1.00                      !node index,pcnsk
```

```
14      -1.00                      !node index,pcnsk
 8      -1.00                      !node index,pcnsk
 3      -1.00                      !node index,pcnsk
11      -1.00                      !node index,pcnsk

 3   !Fine packages failed in N_large zone:      2

 4      -1.00                      !node index,pcnsk
26      -1.00                      !node index,pcnsk
27      -1.00                      !node index,pcnsk

17   !Packages failed in N_large zone:           2

31      -1.00                      !node index,pcnsk
30      -1.00                      !node index,pcnsk
29      -1.00                      !node index,pcnsk
 9      -1.00                      !node index,pcnsk
34      -1.00                      !node index,pcnsk
39      -1.00                      !node index,pcnsk
12      -1.00                      !node index,pcnsk
32      -1.00                      !node index,pcnsk
28      -1.00                      !node index,pcnsk
17      -1.00                      !node index,pcnsk
36      -1.00                      !node index,pcnsk
40      -1.00                      !node index,pcnsk
22      -1.00                      !node index,pcnsk
38      -1.00                      !node index,pcnsk
33      -1.00                      !node index,pcnsk
21      -1.00                      !node index,pcnsk
 2      -1.00                      !node index,pcnsk
```

LIST OF TEST ROUTINE *test_ebs_random.f*

```
      program test_ebs_random

      !routine used to test the EBS random release model

      implicit none

      integer:: numinbuffs, numoutbuffs,idaughter,ithp,ip,num_left
      integer:: j,index_in,index_out,memleft,lns,lns1,tmp_ptindex
      integer:: M_fine, N_large, nodes_left,selected_node,irelease
      integer:: newly_failed,trans_node, in_buffer,M_add_i,M_add_N
      integer:: i,k,n,nodes, nzone, seed, itmp,total_nodes,pseudo_node
      integer:: ith_M, nodes_in_region,nspecies,current_node,temp_node
      integer:: M_failed,istp,nsteps,ioverlap,idrop,N_failed

      integer, allocatable, dimension(:)::ptindex,insnode,M_N_region
      integer, allocatable::M_N_failed_nodes(:),node_list(:),i_fine(:)

      real:: yearsleft,scaleconpart,sumdecayout,sumdecayin
      real:: decayin,decayout,rescale,conpart,conpartd4,adjconpart
      real:: x_fine, y_fine, x_dis, y_dis, xy_dis, xy_least_dis
      real:: xmin,ymin,xmax,ymax,xlen,ylen,begin_time,end_time

      real, allocatable, dimension(:,:):: cord
      real, allocatable, dimension(:):: pcnsk,in,t2sk,t1sk

      real*8:: pinmass,sumpinmass,tmp_pcnsk

      !set up parameters. cli_added 8/11/1999

      real, parameter:: pthresh=0.5, pthresh2=0.25,add0_5=0.5
      real, parameter:: yearindays=365.25, dayinsecs=86400.
      real, parameter:: large_dis=1.E9

      !input parameters for generating coordinates

      open(7,file='nodes')

      print *, 'Input total Number of Nodes'
      read(5,*)total_nodes
      write(7,111)total_nodes
111   format(I7,33x,'!total_nodes, total number of nodes')

      print *,'Input number of sub-regions'
      read(5,*)N_large
      write(7,112)N_large
112   format(I7,33x,'!N_large, number of sub-regions')

      print *, 'Input the seed for the random number geneator'
      read(5,*)seed
      write(7,113)seed
113   format(I7,33x,'!seed, for random number generator')

      print *,'Input total number of M_fine failed packages'
      read(5,*)M_fine
      write(7,114)M_fine
114   format(I7,33x,'!M_fine, number of M_fine failed packages')

      print *,'Input number of species'
      read(5,*)nspecies
      write(7,116)nspecies
116   format(I7,33x,'!nspecies, number of species')

      print *,'Released at this test (->1) or (->0)'
      read(5,*)irelease
      write(7,117)irelease
117   format(I7,33x,'!irelease, Released at this test (->1 or ->0)'/)

      !allocate array and initialize the coordinate arrays

      total_nodes=total_nodes+1
      if(.not.allocated(cord))then
        allocate(cord(total_nodes,2),ptindex(total_nodes),
     &       pcnsk(total_nodes),insnode(total_nodes))
        itmp=9+M_fine*2+N_large+M_fine*N_large
        allocate(in(itmp),t1sk(total_nodes),t2sk(total_nodes))
        allocate(node_list(total_nodes))
      endif

      total_nodes=total_nodes-1
      do i=1,total_nodes
        node_list(i)=i
      enddo

      in(4)=M_fine
      in(5+M_fine*2)=N_large
      in(7+M_fine*2+N_large)=irelease

      cord(1:total_nodes,1)=-1.
      cord(1:total_nodes,2)=-1.

      !generate random node coordinates

      insnode(1)=0
      itmp=0
      current_node=0

      do i=1,N_large
        print *, 'For sub-region: ',i
        print *, 'Input xmin and xmax'
        read(5,*)xmin,xmax
        print *,'Input ymin and ymax'
        read(5,*)ymin,ymax
        print *, 'Input number of nodes in zone',i
        read(5,*)nodes
        write(7,118)i
        write(7,119)xmin,xmax
        write(7,120)ymin,ymax
        write(7,121)nodes,i
118     format(/'!Input parameters for sub-region: ',I7/)
119     format('(',F10.2,',',F10.2,')',17x,'!xmin, xmax, X-range')
```

```
120     format('(',F10.2,',',F10.2,')',17x,'!ymin, ymax, Y-range'/)
121     format(I7,33x,'!number of nodes in zone',I7/)

        xlen=xmax-xmin
        ylen=ymax-ymin

        do n=1,nodes
          num_left=total_nodes-current_node
          pseudo_node=num_left*ran(seed)+0.5
          pseudo_node=current_node+pseudo_node
          if(pseudo_node == current_node)pseudo_node=current_node+1
          current_node=current_node+1
          temp_node=node_list(current_node)
          node_list(current_node)=node_list(pseudo_node)
               node_list(pseudo_node)=temp_node
               pseudo_node=node_list(current_node)
          itmp=itmp+1
          cord(pseudo_node,1)=xmin+xlen*ran(seed)
          cord(pseudo_node,2)=ymin+ylen*ran(seed)
          write(7,122)pseudo_node,cord(pseudo_node,1),
     &                                cord(pseudo_node,2)
122            format(I7,3x,2(F10.2,3x), 4x,'!node_index,(x,y)')
          ptindex(itmp)=pseudo_node
          pcnsk(itmp)=-1
        enddo
        insnode(i+1)=itmp

        print *,'How many M_fine failed package in this region'
        read(5,*)M_failed
        write(7,123)M_failed
123     format(/,I7,33x,'!number of individually failed package')

        do n=1,M_failed
          ith_M=ith_m+1
          if(ith_M <= M_fine)then
            in(5+(ith_M-1)*2)=xmin+xlen*ran(seed)
                 in(4+ith_m*2)=ymin+ylen*ran(seed)
            write(7,124)in(5+(ith_M-1)*2),in(4+ith_M*2)
124              format(2(F10.2,3x),14x'!(x,y) of failed package')
          else
            print *,'Maximum number of M_fine already reached'
            write(7,*)'Maximum number of M_fine already reached'
            exit
          endif
        enddo
      enddo
      print *,'Node coordinates generated'
      write(7,*)''
      write(7,125)
125   format('Node coordinates generated')

      print *,'Input how many time steps to simulate'
      read(5,*)nsteps
      write(7,126)nsteps
126   format(/,I7,33x,'!nsteps, number of time steps'/)

      do istp=1,nsteps
```

```
        print *,'Time step: ', istp
        write(7,127)istp
127     format(/,'At time step:',I5,22x,'!istp, time step'/)
        ith_m=0
        do j=1,N_large
          print *,'Input data for zone ',j
          print *,'Input number of N_failed package'
          read(5,*)N_failed
          write(7,128)N_failed,j
128       format(I7,33x,'!Number of failed packages in sub-region:',I7/)
          in(5+M_fine*2+j)=N_failed
        enddo
        write(7,129)
129     format('!locating node having least dist. to the given (x,y)'/)


      !get injection time in days

      t1sk(1) = begin_time/dayinsecs
      t2sk(1) = end_time/dayinsecs

      !get # of buffer infromation from RIP, cli_08/27/1999

      if(in(7+M_fine*2+N_large) == 0)stop 'No mass will be released'

      !for random ebs release, get M_fine_group and N_large_group #

      M_fine=in(4)
      N_large=in(5+M_fine*2)

      !allocate arrays for EBS random release. cli_added 8/31/1999

      if(.not. allocated(M_N_region))then
        M_add_N=M_fine+N_large
        allocate (M_N_region(M_add_N), M_N_failed_nodes(M_add_N))
        allocate (i_fine(N_large))
        i_fine=0
        M_N_region=0
        M_N_failed_nodes=0
      endif

      !Select release nodes for random EBS release

      !Get M_fine group (x,y) and locate the node index and region

      if(istp ==  1 )then
      in_buffer=4
      do i=1,M_fine
        ioverlap=0
        in_buffer=in_buffer+1
        x_fine=in(in_buffer)
        in_buffer=in_buffer+1
        y_fine=in(in_buffer)
        xy_least_dis=large_dis
        do j=1,N_large
          do k=insnode(j)+1,insnode(j+1)
            x_dis=cord(ptindex(k),1)-x_fine
```

```
                 y_dis=cord(ptindex(k),2)-y_fine
            xy_dis=sqrt(x_dis*x_dis+y_dis*y_dis)
                if(xy_dis < xy_least_dis)then
             M_N_failed_nodes(i)=-k
                  M_N_region(i)=j
                       xy_least_dis=xy_dis
           endif
             enddo
      enddo

      !check for overlap

        do j=1,i-1
          if(M_N_failed_nodes(i) == M_N_failed_nodes(j))then
            ioverlap=ioverlap+1
               endif
        enddo

        if(ioverlap /= 0)then
          print *, 'Found',ioverlap,' overlaped locations'
          write(7,130)
130            format(/'!Found overlaped node'/)
          write(7,132)x_fine,y_fine,ptindex(-M_N_failed_nodes(i)),
     &                                   M_N_region(i)
          write(7,131)
131            format(/'!Program terminated, reassign values'/)
          stop 'Program terminated, reassign values'
        endif


      !shuffle the nodes for M_fine group

        selected_node=insnode(M_N_region(i)+1)-i_fine(M_N_region(i))
        trans_node=-M_N_failed_nodes(i)
        tmp_ptindex=ptindex(selected_node)
        ptindex(selected_node)=ptindex(trans_node)
        ptindex(trans_node)=tmp_ptindex
        tmp_pcnsk=pcnsk(selected_node)
        pcnsk(selected_node)=pcnsk(trans_node)
        pcnsk(trans_node)=tmp_pcnsk
        M_N_failed_nodes(i)=-selected_node
        i_fine(M_N_region(i))=i_fine(M_N_region(i))+1
        write(7,132)x_fine,y_fine,ptindex(-M_N_failed_nodes(i)),
     &                            M_N_region(i)
132     format(2(F10.2),2(I7,3x),'!(x,y),node index,sub-region')

      enddo
      endif

      !For N_large groups: randomly select N_failed package nodes

      in_buffer=5+M_fine*2
      do i=1, N_large
        in_buffer=in_buffer+1
        N_failed=in(in_buffer)
        M_add_i=M_fine+i
        nodes_in_region=insnode(i+1)-insnode(i)-i_fine(i)
        newly_failed=N_failed-M_N_failed_nodes(M_add_i)
        nodes_left=nodes_in_region-M_N_failed_nodes(M_add_i)
        if(N_failed < nodes_in_region)then
          M_N_region(M_add_i)=i
          do j=1,newly_failed
            if(newly_failed > 1 )then
              do
                k=int(ran(seed)*nodes_left+add0_5)
                         if(k /= 0)exit
                   enddo
                 else
              k=1
            endif
            selected_node=insnode(i)+M_N_failed_nodes(M_add_i)+k
                 trans_node=insnode(i)+M_N_failed_nodes(M_add_i)+1
            tmp_ptindex=ptindex(trans_node)
            ptindex(trans_node)=ptindex(selected_node)
            ptindex(selected_node)=tmp_ptindex
            tmp_pcnsk=pcnsk(trans_node)
            pcnsk(trans_node)=pcnsk(selected_node)
            pcnsk(selected_node)=tmp_pcnsk
                 nodes_left=nodes_left-1
                 M_N_failed_nodes(M_add_i)=M_N_failed_nodes(M_add_i)+1
          enddo
        else
          M_N_failed_nodes(M_add_i)=nodes_in_region
          M_N_region(M_add_i)=i
        endif

      !output results

        write(7,135)i_fine(i),i
135     format(/,I7,3x,'!Fine packages failed in N_large zone:',I7/)
        itmp=insnode(i+1)
        do n=1,i_fine(i)
          write(7,136)ptindex(itmp),pcnsk(itmp)
136            format(I7,5x,F6.2,22x,'!node index,pcnsk')
          itmp=itmp-1
        enddo
        write(7,137)M_N_failed_nodes(M_add_i),i
137     format(/,I7,3x,'!Packages failed in N_large zone:',5x,I7/)
        do n=insnode(i)+1,(insnode(i)+M_N_failed_nodes(M_add_i))
          write(7,136)ptindex(n),pcnsk(n)
        enddo

      enddo

      enddo

      end program test_ebs_random
```